SPIKING NEURAL NETWORKS: A THEORETICAL FRAMEWORK FOR UNIVERSAL APPROXIMATION AND TRAINING

UMBERTO BICCARI*

ABSTRACT. Spiking Neural Networks (SNNs) are widely regarded as a biologically inspired and energy-efficient alternative to classical artificial neural networks. Yet, their theoretical foundations remain only partially understood. In this work, we develop a rigorous mathematical analysis of a representative SNN architecture based on Leaky Integrate-and-Fire (LIF) neurons with threshold-reset dynamics. Our contributions are twofold. First, we establish a universal approximation theorem showing that SNNs can approximate continuous functions on compact domains to arbitrary accuracy. The proof relies on a constructive encoding of target values via spike timing and a careful interplay between idealized δ-driven dynamics and smooth Gaussian-regularized models. Second, we analyze the qualitative behavior of spike times across layers, proving well-posedness of the hybrid dynamics and deriving conditions under which spike counts remain stable, decrease, or in exceptional cases increase due to resonance phenomena or overlapping inputs. Together, these results provide a principled foundation for understanding both the expressive power and the dynamical constraints of SNNs, offering theoretical guarantees for their use in classification and signal processing tasks.

1. INTRODUCTION

Spiking Neural Networks (SNNs) have emerged as a promising paradigm for developing energy efficient and biologically plausible models of computation. Unlike classical Artificial Neural Networks (ANNs), which process continuous-valued activations, SNNs transmit information through discrete spikes, closely emulating the signaling mechanisms of real neurons. This event-driven nature makes them particularly appealing for applications in neuromorphic hardware and real-time sensory processing, where sparse activity and low energy consumption are crucial [8, 10, 22, 26]. At the same time, SNNs offer a conceptual bridge between neuroscience and machine learning, providing models that are both biologically plausible and computationally powerful.

Despite their conceptual appeal, training SNNs remains challenging due to the non-differentiable nature of the spiking mechanism and the inherent hybrid dynamics governing neuronal behavior. Recent advances [8] have sought to bridge this gap by importing ideas from deep learning and control theory to derive principled training algorithms that respect the underlying neurodynamics. Yet, the theoretical foundations of SNNs remain less developed than those of ANNs. Two fundamental questions, central to understanding their capabilities, are still only partially resolved.

The first concerns expressive power: to what extent can SNNs approximate arbitrary nonlinear functions? Early milestones [21, 22] demonstrated that spiking networks are computationally as powerful as Turing machines, and subsequent works explored their approximation capabilities in both rate-based and temporal coding frameworks [9, 25, 29]. More recently, surveys such as [4, 24] and methodological advances like [8] have reinforced the role of SNNs within modern deep learning. Yet, most of these results rely on heuristic or asymptotic arguments, leaving a gap for constructive, mathematically rigorous proofs of universal approximation in realistic spiking models.

The second question relates to spike dynamics: how do the hybrid dynamical laws governing spiking neurons constrain or facilitate information transmission across layers? Although the dynamics of Leaky Integrate-and-Fire (LIF) and Hodgkin–Huxley neurons are well understood in computational neuroscience [6,13], there is still limited mathematical analysis of spike counts, their stability across layers, and the exceptional regimes – such as resonance phenomena or overlapping inputs – where they may increase. A deeper understanding of these mechanisms is crucial, since the expressive power of SNNs cannot be decoupled from the dynamical rules that govern their activity.

In this work, we contribute to filling both gaps. Our first main result is a constructive Universal Approximation Theorem for SNNs. We prove that SNNs built from LIF neurons with threshold-reset dynamics can approximate any continuous function on compact domains with arbitrary accuracy. The proof explicitly encodes target function values via spike timing, relying on a careful interplay between an idealized δ-driven model[1] – used for constructive encoding – and a Gaussian-regularized model, which ensures well-posedness and differentiability. This dual modeling strategy is essential: the δ-formulation provides the conceptual clarity needed for constructive analysis, while the regularized version ensures well-posed and practically implementable dynamics.

Our second contribution is a qualitative analysis of spike-time propagation across layers. We establish the well-posedness of the hybrid dynamical systems defining membrane potentials, and we identify conditions under which spike counts remain stable, decrease, or exceptionally increase. In particular, we show that under natural assumptions, spike counts form a non-increasing sequence across layers, with precise criteria describing the rare cases – such as overlapping presynaptic spikes – that may lead to increases. This analysis provides a rigorous mathematical explanation of how SNNs' expressive power interacts with their dynamical constraints.

These two perspectives complement one another: the universal approximation theorem highlights the expressive capacity of SNNs, while the qualitative study of spike times elucidates the dynamical mechanisms through which such capacity is realized or limited. Together, they provide a rigorous mathematical foundation for understanding the power and constraints of SNNs and deliver a principled theoretical foundation for their study. By combining a constructive universal approximation framework with a detailed dynamical analysis, we bridge the gap between the expressive potential of spiking architectures and the constraints imposed by their hybrid dynamics. In doing so, we complement recent empirical and algorithmic advances, offering rigorous guarantees that support the use of SNNs in classification, signal processing, and beyond.

This paper is organized as follows. In Section 2, we introduce the general architecture of the SNN under consideration, describing in detail the input, hidden, output, and readout layers together with their governing dynamics. Section 3 is devoted to the well-posedness analysis of these hybrid dynamical systems and to the characterization of spike times across layers, providing conditions for stability and growth of spike counts. In Section 4, we present and prove our main result: a constructive Universal Approximation Theorem for SNNs, highlighting the interplay between δ-driven and Gaussian-regularized dynamics. Section 5 develops a principled training framework based on surrogate gradients and optimal control methods, which enables gradient-based optimization of the proposed architecture. Section 6 reports simulation experiments that validate the theoretical findings and illustrate the practical behavior of the proposed framework. Finally, Section 7 summarizes our contributions and outlines open questions for future research.

2. General framework

As anticipated, SNNs process information in a fundamentally different way from classical ANNs. Rather than working with continuous activations, they rely on discrete events (spikes) that occur in time. These spikes serve as the basic units of communication between neurons, giving rise to

---

[1]Here, δ-driven means that the model's formulation involves Dirac δ distribution.

dynamics that are both time-dependent and event-driven. This temporal nature makes SNNs closer to biological computation and motivates the mathematical framework we now introduce.

Consider a dataset $\mathcal{D} := \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{N}$ with $\boldsymbol{x}_i \in \mathbb{R}^d$ and $y_i \in \{0, 1\}$ for all $i \in [\![N]\!] := \{1, \ldots, N\}$. We want to classify $\mathcal{D}$ employing SNNs. In particular, let us focus on the following architecture (see Figure 2.1):

1. One input layer $\mathcal{L}^{\text{in}}$ with one neuron $\mathcal{N}^{\text{in}}$ whose behavior is described by a **membrane potential**

$$v[\boldsymbol{x}_i](t) : \mathbb{R} \to \mathbb{R}, \quad \text{for all } t \in (0, T),$$

   receiving as input the data $\{\boldsymbol{x}_i\}_{i=1}^{N}$.
2. $L \geq 1$ hidden layers $\{\mathcal{L}_{\ell}^{\text{hidd}}\}_{\ell=1}^{L}$, each one of them with $P \geq 1$ neurons $\{\mathcal{N}_{\ell,p}^{\text{hidd}}\}_{p=1}^{P}$ with membrane potential $\xi_{\ell,p}$. Each of these hidden layers receives as input the output of the layer immediately previous to it, that is,

$$\begin{aligned} \boldsymbol{\xi}_1 &= \boldsymbol{\xi}_1[v](t) : \mathbb{R} \to \mathbb{R}, && \text{for all } t \in (0, T), \\ \boldsymbol{\xi}_{\ell+1} &= \boldsymbol{\xi}_{\ell+1}[\boldsymbol{\xi}_\ell](t) : \mathbb{R} \to \mathbb{R}, && \text{for all } t \in (0, T) \text{ and } \ell \in [\![L-1]\!], \end{aligned}$$

   where for all $\ell \in [\![L]\!]$, $\boldsymbol{\xi}_\ell = (\xi_{\ell,1}, \ldots, \xi_{\ell,P}) \in \mathbb{R}^P$ denotes the vector collecting the P membrane potentials of $\mathcal{L}_{\ell}^{\text{hidd}}$.
3. One output layer $\mathcal{L}^{\text{out}}$ composed of P neurons $\{\mathcal{N}_{p}^{\text{out}}\}_{p=1}^{P}$ with membrane potentials

$$u_p[\xi_{L,p}](t) : \mathbb{R} \to \mathbb{R}, \quad \text{for all } t \in (0, T) \text{ and } p \in [\![P]\!],$$

   each one of them receiving information of the p-th neuron in the last hidden layer.
4. A final readout layer $\mathcal{L}^{\text{read}}$ with one readout neuron $\mathcal{N}^{\text{read}}$ aggregating the outputs of $\{\mathcal{N}_{p}^{\text{out}}\}_{p=1}^{P}$ and performing classification.
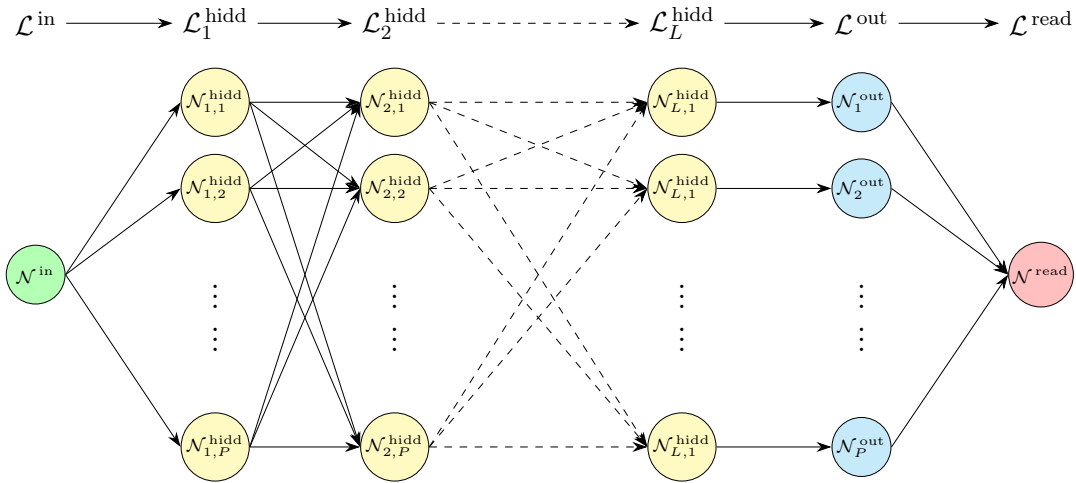

FIGURE 2.1. Scheme of the employed SNN architecture. The network is organized into an input layer, several hidden layers, and an output layer. Information is transmitted forward through discrete spike events, and the final layer provides the features used for prediction.

2.1. **Modeling of the membrane potentials.** For the membrane potentials we will employ the *Leaky Integrate-and-Fire* (LIF) model, which consists in modeling each neuron as a low-pass filter circuit with one resistor and one capacitor. This modeling is applied uniformly across the network: input, hidden, and output layers are all described by LIF-type dynamics.

From a biological perspective, the LIF model distills key mechanisms of neuronal behavior into a mathematically tractable form. In real neurons, the membrane potential arises from the flow of ions across the cell membrane through specialized channels, while incoming signals arrive as synaptic currents that transiently alter this potential [13, 16]. When the accumulated depolarization crosses a critical threshold, the neuron emits an action potential (a spike), after which ion pumps and channel dynamics restore the potential to its resting state [20]. The LIF model abstracts this process by representing the membrane as an electrical RC circuit: the capacitor mimics the ability of the membrane to store charge, the resistor accounts for the natural leakage of ions through open channels, and synaptic inputs are modeled as external currents charging the circuit. Spiking then occurs when the voltage across this circuit exceeds a prescribed threshold, triggering an instantaneous reset that mirrors the refractory recovery of biological neurons [6, 10, 11, 14].
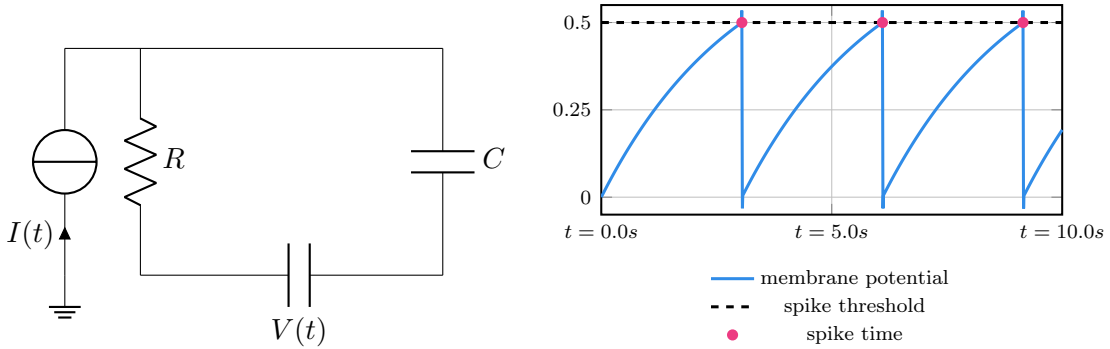

FIGURE 2.2. General scheme of the LIF model. Left: the RC circuit representation, where the input current $I(t)$ charges the membrane potential $V(t)$. Right: an example of the resulting spiking behavior when $V(t)$ crosses a threshold and is reset. In Sections 2.1.1, 2.1.2 and 2.1.3, the specific forms of $I(t)$ and $V(t)$ will be detailed for the input, hidden, and output layers of the SNN.

In our architecture, we adopt this LIF abstraction uniformly across all layers, but with a distinction motivated by function: in the input and hidden layers we retain the threshold–reset mechanism to reflect the discrete, event-driven nature of biological spiking, whereas in the output layer we omit the reset so that the potentials evolve continuously and provide stable features for the final static readout. This design preserves the biological inspiration of spike-based communication while ensuring a mathematically well-posed framework suitable for universal approximation.

We now make these modeling choices precise by writing down the dynamical systems that govern the evolution of the membrane potentials in the input, hidden, and output layers, as well as the readout ansatz. The analysis of well-posedness for the LIF models and the characterization of spike times are deferred to Section 3.

2.1.1. *Input layer.* Let $(0, T)$ with $T > 0$ be a given time interval, $\theta_v > 0$ denote the spike threshold for $\mathcal{N}^{\text{in}}$ and set $\tau_v := R_v C_v$, with (R_v, C_v) denoting the neuron's resistor and capacitor. For all $i \in [\![N]\!]$, the presynaptic neuron's potential $v = v[\boldsymbol{x}_i]$ evolves according to the hybrid dynamical

system

$$\begin{cases} \dot{v}(t) = \dfrac{1}{\tau_v}\Big(-v(t) + \langle \boldsymbol{a}, \boldsymbol{x}_i \rangle \Big), & t \in (0,T) \setminus \mathbb{T}_0 \\ v(0) = 0 \\ v_k^- = \theta_v, \quad v_k^+ = 0 \end{cases} \tag{In}$$

where $\boldsymbol{a} \in \mathbb{R}^d$ is a trainable parameter modulating the input strength,

$$\mathbb{T}_0 = \Big\{ t_{0,k} \in (0,T) \ : \ k \in \mathbb{N} \text{ and } v(t_{0,k}) = \theta_v \Big\}$$

denotes the sequence of spike times, and

$$v_k^\pm := \lim_{t \to t_{0,k}^\pm} v(t).$$

2.1.2. *Hidden layers.* For all $\ell \in [\![L]\!]$ and $p \in [\![P]\!]$, let $\theta_{\ell,p} > 0$ denote the spike threshold for the neuron $\mathcal{N}_{\ell,p}^{\text{hidd}}$ and set $\tau_{\ell,p} := R_{\ell,p} C_{\ell,p}$, with $(R_{\ell,p}, C_{\ell,p})$ denoting the neuron's resistor and capacitor. The hidden layers neurons' potential $\xi_{\ell,p}$ evolves according to the hybrid dynamical system

$$\begin{cases} \dot{\xi}_{\ell,p}(t) = \dfrac{1}{\tau_{\ell,p}}\Big(-\xi_{\ell,p}(t) + \omega_{\ell,p} J_\ell(t) \Big), & t \in (0,T) \setminus \mathcal{T}_{\ell,p} \\ \xi_{\ell,p}(0) = 0 \\ \xi_{\ell,p,k}^- = \theta_{\ell,p}, \quad \xi_{\ell,p,k}^+ = 0 \end{cases} \tag{Hidd}$$

where $\boldsymbol{\omega}_\ell = (\omega_{\ell,1}, \ldots \omega_{\ell,P}) \subset \mathbb{R}^P$ are trainable parameters,

$$\mathcal{T}_{\ell,p} = \Big\{ t_{\ell,p,k} \in (0,T) \ : \ k \in \mathbb{N} \text{ and } \xi_{\ell,p}(t_{\ell,p,k}) = \theta_{\ell,p} \Big\}$$

denotes the sequence of spike times,

$$\xi_{\ell,p,k}^\pm := \lim_{t \to t_{\ell,p,k}^\pm} \xi_{\ell,p}(t),$$

and where we have defined the input current

$$J_\ell(t) := \sum_{t^* \in \mathbb{T}_{\ell-1}} \mathcal{G}_{\mu,t^*}(t), \qquad \text{for all } \ell \in [\![L]\!], \tag{2.1}$$

with

$$\mathcal{G}_{\mu,t^*}(t) := \frac{1}{\mu\sqrt{2\pi}} e^{-\frac{(t-t^*)^2}{2\mu^2}}$$

a Gaussian distribution centered at t^* with amplitude $\mu > 0$ and where

$$\mathbb{T}_{\ell-1} = \bigcup_{p=1}^{P} \mathcal{T}_{\ell-1,p}$$

collects the spike times generated by all neurons in the $(\ell-1)$-th hidden layer (for $\ell > 1$) or, in the case $\ell = 1$, by the input neuron.

5

Remark 2.1. In a typical SNN architecture, information among the neurons/layers is transmitted through **spikes**. To describe this phenomenon, one should use the following hybrid ODEs

$$
\begin{cases}
\dot{\xi}_{\ell,p}(t) = \dfrac{1}{\tau_{\ell,p}}\left(-\xi_{\ell,p}(t) + \omega_{\ell,p}\displaystyle\sum_{t^* \in \mathbb{T}_{\ell-1}} \delta(t-t^*)\right), & t \in (0,T) \setminus \mathcal{T}_{\ell,p} \\
\xi_{\ell,p}(0) = 0 \\
\xi^-_{\ell,p,k} = \theta_{\ell,p}, \quad \xi^+_{\ell,p,k} = 0
\end{cases}
, \tag{2.2}
$$

which models more faithfully the impulsive nature of spike transmission. However, due to the presence of the Dirac deltas, (2.2) is not well-posed in standard ODE theory: the right-hand side is only defined in the sense of distributions, and solutions are not classical trajectories [15].

For this reason, for the rigorous analysis in this work, we have decided to focus on models like (Hidd) where each delta $\delta(t-t^*)$ is replaced by a smooth Gaussian kernel $G_{\mu,t^*}(t)$. This mollification process is of course motivated by the weak-$*$ convergence of $\mathcal{G}_{\mu,t^*}(t)$ to $\delta(t-t^*)$: for any test function $\varphi \in C_0^\infty(\mathbb{R})$, we have

$$
\lim_{\mu\to 0^+} \int_{\mathbb{R}} G_{\mu,t^*}(t)\varphi(t)\,dt = \varphi(t^*) = \int_{\mathbb{R}} \delta(t-t^*)\varphi(t)\,dt.
$$

At the same time, notice that one may still write a natural *formal* solution of (2.2) by interpreting each delta impulse as producing an instantaneous exponential response in the membrane potential. Concretely, if $\{t^*\}$ are the input spike times, one obtains a symbolic expression of the form

$$
\xi_{\ell,p,\delta}(t) = \frac{\omega_{\ell,p}}{\tau_{\ell,p}}\sum_{t^* \in \mathbb{T}_{\ell-1}} e^{-\frac{t-t^*}{\tau_{\ell,p}}},
$$

interleaved with threshold-triggered resets. This explicit representation, though heuristic, will later be exploited in the proof of Theorem 4.1 to construct spike trains that encode prescribed target values.

In summary, throughout the paper, the delta formulation serves as an idealized proxy for intuition and constructive arguments, while the Gaussian model underpins the rigorous analysis and training framework.

2.1.3. *Output layer.* Also the postsynaptic neurons $\{\mathcal{N}_p^{\text{out}}\}_{p=1}^P$ will be modeled as an RC circuit, receiving as inputs the spiking information from the last hidden layer's neurons $\{\mathcal{N}_{L,p}^{\text{hidd}}\}_{p=1}^P$. In particular, we will consider the dynamical systems

$$
\begin{cases}
\dot{u}_p(t) = \dfrac{1}{\tau_u}\left(-u_p(t) + w\Phi_p(t)\right) & t \in (0,T) \\
u_p(0) = 0
\end{cases}
, \qquad \text{for all } p \in [\![P]\!], \tag{Out}
$$

with $\tau_u > 0$, input current

$$
\Phi_p(t) := \sum_{t^* \in \mathbb{T}_{L,p}} \mathcal{G}_{\mu,t^*}(t), \tag{2.3}
$$

and $w \in \mathbb{R}$ a trainable parameter common to each postsynaptic neuron. Also in this case, (Out) is a smooth approximation of

$$
\begin{cases}
\dot{u}_p(t) = \dfrac{1}{\tau_u}\left(-u_p(t) + w\displaystyle\sum_{t^* \in \mathbb{T}_{L,p}} \delta(t-t^*)\right), & t \in (0,T) \\
u_p(0) = 0
\end{cases}
, \tag{2.4}
$$

that we introduce to simplify the analysis.

6

2.1.4. *Readout layer.* The final stage of the network is a static readout layer, acting on the membrane potentials produced by the output neurons at the final time T. Unlike the preceding layers, this component does not evolve dynamically but instead implements a fixed functional of the form

$$\mathcal{R}(\boldsymbol{u}(T)) := \sum_{p=1}^{P} \nu_p \sigma\big(u_p(T) - \theta_u\big),$$

where $\boldsymbol{u} = (u_1, \ldots, u_P)$ denotes the vector of output potentials, $\theta_u > 0$ is a given threshold, $\sigma : \mathbb{R} \to \mathbb{R}$ is a Lipschitz non-polynomial activation function, and $\boldsymbol{\nu} = (\nu_1, \ldots, \nu_P) \in \mathbb{R}^P$ are trainable parameters.

This readout realizes a Cybenko-type ansatz [19]: the continuous features $u_p(T)$ generated by the dynamical system are non-linearly combined with trainable weights to produce the final network output. The choice of this structure is crucial, since it ensures the universal approximation property of the architecture that will be established in Section 4.

2.1.5. *Full SNN mapping.* Combining the dynamical evolution of the input, hidden, and output layers with the static readout functional, we obtain a mapping from the input space $\mathbb{R}^d$ to the real line. For an input datum $x \in \mathbb{R}^d$, let

$$v(t, \boldsymbol{x}) \in \mathbb{R}, \quad \{\boldsymbol{\xi}_\ell(t, \boldsymbol{x})\}_{\ell=1}^{L} \subset \mathbb{R}^P, \quad \boldsymbol{u}(t, \boldsymbol{x}) \in \mathbb{R}^P$$

denote the membrane potentials of the input, hidden, and output layers, respectively, at time t, governed by the dynamical systems specified in Subsections 2.1.1-2.1.2-2.1.3. The network output corresponding to $\boldsymbol{x}$ is then defined by applying the readout functional of Subsection 2.1.4:

$$\mathrm{SNN}(\boldsymbol{x}) := \mathcal{R}(\boldsymbol{u}(T, \boldsymbol{x})) = \sum_{p=1}^{P} \nu_p \sigma\big(u_p(T, \boldsymbol{x}) - \theta_u\big). \tag{2.5}$$

This mapping $\mathrm{SNN} : \mathbb{R}^d \to \mathbb{R}$ will serve as the fundamental ansatz of the model. In Section 4 we will show, in the spirit of Cybenko's analysis [5, 19], that the class of such mappings possesses the universal approximation property.

Finally, we emphasize that the explicit dependence on $\boldsymbol{x}$ is shown here for the sake of clarity, in order to make the network mapping precise. In the dynamical systems of Subsections 2.1.1-2.1.2-2.1.3 this dependence was left implicit, and in Section 3 it will again be suppressed in the notation so as to focus on the dynamical behavior of the LIF models themselves.

3. Well-posedness analysis

We discuss here the well-posedness and some elementary properties of the dynamical systems describing the evolution of the neurons membranes' potentials. Our main interest will be to characterize the spike times at which information is transmitted from one layer to the next one.

3.1. **Input layer.** To set the stage for our well-posedness analysis, we first examine the input layer, where the emergence of spike times can be described explicitly.

Proposition 3.1. *For all $\boldsymbol{x}_i \in \mathbb{R}^d$, $\boldsymbol{a} \in \mathbb{R}^d$, $0 < \tau_v, \theta_v \in \mathbb{R}$ and $T > 0$ there exists a unique solution v of the dynamical system* (In) *such that*

$$v \in C^\infty\big((0, T) \setminus \mathbb{T}_0\big),$$

where $\mathbb{T}_0$ denotes the sequence of spike times $\{t_{0,k}\} \subset (0, T)$ at which $v(t_{0,k}) = \theta_v$. Moreover, $\mathbb{T}_0$ can be characterized as follows:

1. *if $\langle \boldsymbol{a}, \boldsymbol{x}_i \rangle \leq \theta_v$, then $\mathbb{T}_0 = \varnothing$;*

2. *if $\langle \boldsymbol{a}, \boldsymbol{x}_i \rangle > \theta_v$, let us define*

$$\beta(\boldsymbol{a}, \boldsymbol{x}_i) := \tau_v \ln \left(\frac{\langle \boldsymbol{a}, \boldsymbol{x}_i \rangle}{\langle \boldsymbol{a}, \boldsymbol{x}_i \rangle - \theta_v} \right) \tag{3.1}$$

and set $K := \lfloor T\beta^{-1} \rfloor$. Then

$$\mathbb{T}_0 = \begin{cases} \varnothing & \text{if } K = 0 \\ \{t_{0,k}\}_{k=1}^K \text{ with } t_{0,k} := k\beta(a) & \text{if } K \geq 1. \end{cases} \tag{3.2}$$

In particular, the spike times are either the empty set or a growing periodic sequence with period β.

Proof. First of all, notice that the solution to (In) can be computed explicitly, by superposing the solution of

$$\begin{cases} \dot{v}(t) = \frac{1}{\tau_v}\Big(-v(t) + \langle \boldsymbol{a}, \boldsymbol{x}_i \rangle \Big), & t \in (0, T) \\ v(0) = 0 \end{cases},$$

which is given by

$$v(t) = \langle \boldsymbol{a}, \boldsymbol{x}_i \rangle \left(1 - e^{-\frac{t}{\tau_v}} \right), \tag{3.3}$$

with the reset mechanism

$$v_k^- = \lim_{t \to t_{0,k}^-} v(t) = \theta_v, \qquad v_k^+ = \lim_{t \to t_{0,k}^+} v(t) = v_k^- - \theta_v = 0. \tag{3.4}$$

This reset mechanism (3.4) is activated whenever v reaches the threshold θ_v during the time interval $(0, T)$. This happens or not, depending on the input $\langle \boldsymbol{a}, \boldsymbol{x}_i \rangle$.

Case 1: $\langle \boldsymbol{a}, \boldsymbol{x}_i \rangle \leq 0$. In this case, the solution (3.3) is always constant or decreasing in time. Together with $v(0) = 0$, this implies that $v(t) \leq 0$ for all $t \in (0, T)$. Since $\theta_v > 0$, we therefore have that $\mathbb{T}_0 = \varnothing$ and $v \in C^\infty(0, T)$.

Case 2: $0 < \langle \boldsymbol{a}, \boldsymbol{x}_i \rangle \leq \theta_v$. In this case, the solution (3.3) is increasing in time and fulfills the bound

$$0 \leq v(t) < \langle \boldsymbol{a}, \boldsymbol{x}_i \rangle, \quad \text{for all } t \in (0, T).$$

This in particular implies that $v(t) < \theta_v$ for all $t \in (0, T)$. Hence, we have again that $\mathbb{T}_0 = \varnothing$ and $v \in C^\infty(0, T)$.

Case 3: $\langle \boldsymbol{a}, \boldsymbol{x}_i \rangle > \theta_v$. Suppose now that $\langle \boldsymbol{a}, \boldsymbol{x}_i \rangle > \theta_v$. In this case, v grows from $v(0) = 0$ and reaches for the first time the threshold θ_v at a certain $t_{0,1} > 0$, which can be computed explicitly by imposing in (3.3) that $v(t_{0,1}) = \theta_v$. This gives

$$t_{0,1} = \tau_v \ln \left(\frac{\langle \boldsymbol{a}, \boldsymbol{x}_i \rangle}{\langle \boldsymbol{a}, \boldsymbol{x}_i \rangle - \theta_v} \right) = \beta(\boldsymbol{a}, \boldsymbol{x}_i).$$

Notice that, since $\langle \boldsymbol{a}, \boldsymbol{x}_i \rangle > \theta_v$, this value $t_{0,1}$ is defined and strictly positive. Now, if $\beta > T$, i.e. if $K = \lfloor T\beta^{-1} \rfloor = 0$, then $t_{0,1} \notin (0, T)$. Hence, once again, we have that $\mathbb{T}_0 = \varnothing$ and $v \in C^\infty(0, T)$. Otherwise, we can split

$$(0, T) = (0, t_{0,1}) \cup \{t_{0,1}\} \cup (t_{0,1}, T)$$

and the solution of (In) is given by

$$v(t) = \langle \boldsymbol{a}, \boldsymbol{x}_i \rangle \left(1 - e^{-\frac{t}{\tau_v}} \right), \qquad \text{if } t \in (0, t_{0,1})$$

$$v(t_{0,1}^-) = \theta_v, \quad v(t_{0,1}^+) = 0 \tag{3.5}$$

$$v(t) = \langle \boldsymbol{a}, \boldsymbol{x}_i \rangle \left(1 - e^{-\frac{t-t_{0,1}}{\tau_v}} \right), \quad \text{if } t \in (t_{0,1}, T)$$

Over the time interval $(t_{0,1}, T)$, this solution (3.5) grows again from $v(t_{0,1}^+) = 0$ and reaches for the second time the threshold θ_v at a certain $t_{0,2} > t_{0,1}$, which can be once again computed explicitly by imposing in (3.5) that $v(t_{0,2}) = \theta_v$. This gives

$$t_{0,2} = t_{0,1} + \tau_v \ln \left(\frac{\langle \boldsymbol{a}, \boldsymbol{x}_i \rangle}{\langle \boldsymbol{a}, \boldsymbol{x}_i \rangle - \theta_v} \right) = t_{0,1} + \beta(\boldsymbol{a}, \boldsymbol{x}_i) = 2\beta(\boldsymbol{a}, \boldsymbol{x}_i).$$

Clearly, this second spike time $t_{0,2}$ will belong to $\mathbb{T}_0$ if and only if $2\beta < T$. This reasoning can be repeated inductively so to get that, for a given $k \in \mathbb{N}^*$,

$$t_{0,k} = k\beta(\boldsymbol{a}, \boldsymbol{x}_i),$$

as long as $k\beta < T$, that is, as long as $k \leq K = \lfloor T\beta^{-1} \rfloor$. The characterization (3.2) of $\mathbb{T}_0$ is therefore complete. Moreover, by construction, we immediately see that

$$v \in C^\infty\big((0, T) \setminus \mathbb{T}_0\big).$$

$\square$

3.2. **Hidden layers.** We analyze here the well-posedness of the hidden layers' dynamical systems (Hidd), and provide a characterization of the spiking times.

Proposition 3.2. *For all $\ell \in [\![L]\!]$, $p \in [\![P]\!]$, $0 < \tau_{\ell,p}, \theta_{\ell,p} \in \mathbb{R}$ and $T > 0$ there exists a unique solution $\xi_{\ell,p}$ of the dynamical system (Hidd) with J_ℓ as in (2.1) such that*

$$\xi_{\ell,p} \in C^\infty\big((0, T) \setminus \mathcal{T}_{\ell,p}\big),$$

where $\mathcal{T}_{\ell,p}$ denotes the sequence of spike times $\{t_{\ell,p,k}\} \subset (0, T)$ at which $\xi_{\ell,p}(t_{\ell,p,k}) = \theta_{\ell,p}$. Moreover, $\mathcal{T}_{\ell,p}$ can be characterized as follows:

1. *if $\mathbb{T}_{\ell-1} = \cup_{p=1}^P \mathcal{T}_{\ell-1,p} = \varnothing$, then $\mathcal{T}_{\ell,p} = \varnothing$;*
2. *if $\mathbb{T}_{\ell-1} \neq \varnothing$, then every spike time $t_{\ell,p,k} \in \mathcal{T}_{\ell,p}$ satisfies the necessary condition*

$$\omega_\ell J_\ell(t_{\ell,p,k}) = \theta_{\ell,p}. \tag{3.6}$$

Proof. First of all, if $\mathbb{T}_{\ell-1} = \varnothing$, we have from (2.1) that $J_\ell(t) \equiv 0$. This clearly implies that $\xi_{\ell,p}(t) \equiv 0$.

Let us now assume that $\mathbb{T}_{\ell-1} \neq \varnothing$. In the same spirit of the proof of Proposition 3.1, notice that the solution to (Hidd) can be computed explicitly, by gluing together the solutions of

$$\begin{cases} \dot{\xi}_{\ell,p}(t) = \dfrac{1}{\tau_{\ell,p}}\big(-\xi_{\ell,p}(t) + \omega_\ell J_\ell(t) \big), & t \in (t_{\ell,p,k-1}, t_{\ell,p,k}) \\ \xi_{\ell,p}(t_{\ell,p,k-1}) = 0 \end{cases} \tag{3.7}$$

and the reset mechanism

$$\xi_{\ell,p,k}^- = \lim_{t \to t_{\ell,p,k}^-} \xi_{\ell,p}(t) = \theta_{\ell,p}, \qquad \xi_{\ell,p,k}^+ = \lim_{t \to t_{\ell,p,k}^+} \xi_{\ell,p}(t) = 0,$$

9

where we have set $t_{\ell,p,0} = 0$. Thanks to the variation of constants formula, the solution of (3.7) is given by

$$\xi_{\ell,p}(t) = \frac{\omega_{\ell,p}}{\tau_{\ell,p}} e^{-\frac{t}{\tau_{\ell,p}}} \int_{t_{\ell,p,k-1}}^{t} e^{\frac{s}{\tau_{\ell,p}}} J_\ell(s) \, ds. \tag{3.8}$$

Moreover, since by construction $J_\ell \in C^\infty(0,T)$, we have that $\xi_{\ell,p} \in C^\infty\big(t_{\ell,p,k-1}, t_{\ell,p,k}\big)$. Finally, the spike times can be computed by imposing in (3.8) that $\xi_{\ell,p}(t_{\ell,p,k}) = \theta_{\ell,p}$, that is

$$\int_{t_{\ell,p,k-1}}^{t_{\ell,p,k}} e^{\frac{s}{\tau_{\ell,p}}} J_\ell(s) \, ds = \frac{\theta_{\ell,p}\tau_{\ell,p}}{\omega_{\ell,p}} e^{\frac{t_{\ell,p,k}}{\tau_{\ell,p}}}.$$

Differentiating this last expression with respect to $t_{\ell,p,k}$ we get the identity

$$e^{\frac{t_{\ell,p,k}}{\tau_{\ell,p}}} J_\ell(t_{\ell,p,k}) = \frac{\theta_{\ell,p}}{\omega_{\ell,p}} e^{\frac{t_{\ell,p,k}}{\tau_{\ell,p}}}.$$

We then see immediately that, if $\xi_{\ell,p}(t_{\ell,p,k}) = \theta_{\ell,p}$, we also have $\omega_\ell J_\ell(t_{\ell,p,k}) = \theta_{\ell,p}$. Our proof is therefore completed. $\qquad\square$

Remark 3.3. Notice that (3.6) provides a set of **candidate times** for spiking. However, not every solution of (3.6) corresponds to an actual spike: a candidate time becomes a true spike time only if the membrane potential $\xi_{\ell,p}(t)$, evolving according to (Hidd) and subject to reset, actually reaches the threshold $\theta_{\ell,p}$ at that instant. Thus, (3.6) is a necessary (but not sufficient) condition characterizing all spike times, in the sense that each spike time belongs to the solution set of (3.6) but the converse inclusion need not hold.

3.3. **Analysis of the spike times' trend.** We analyze here the spike times' trend among the hidden layers $\{\mathcal{L}_\ell^{\text{hidd}}\}_{\ell=1}^L$. In particular, we are interested in determining under which conditions the number of spike times increases or decreases from one layer to the successive one.

Proposition 3.4. *Fix $\ell \in [\![L]\!]$, $p \in [\![P]\!]$, and let $\xi_{\ell,p}(t)$ be the membrane potential of a hidden-layer neuron governed by the hybrid system* (Hidd)*, with input spike times $\mathbb{T}_{\ell-1} = \{t_{\ell-1,k}\}_{k=1}^K \subset (0,T)$ and input current $J_\ell(t)$ given by* (2.1)*. Given $\mu > 0$, suppose there exists $\Delta > 6\mu$ such that the presynaptic spike times $\mathbb{T}_{\ell-1}$ satisfy*

$$t_{\ell-1,k+1} - t_{\ell-1,k} \geq \Delta, \quad \text{for all } k \in [\![K-1]\!]. \tag{3.9}$$

Assume further that the parameter $\omega_{\ell,p}$ and the threshold $\theta_{\ell,p}$ are such that

$$\omega_{\ell,p}\mathcal{G}_{\mu,t_{\ell-1,k}} > \theta_{\ell,p}, \quad \text{for all } k \in [\![K]\!]. \tag{3.10}$$

Then, it holds the following

1. *For each presynaptic spike $t_{\ell-1,k}$ there exists at least one postsynaptic spike $t_{\ell,p,j} \in \mathcal{T}_{\ell,p}$ satisfying*

$$t_{\ell,p,j} \in (t_{\ell-1,k} - \zeta, t_{\ell-1,k}) \quad \text{with} \quad \zeta := \mu\sqrt{2\log\left(\frac{\omega_\ell}{\mu\sqrt{2\pi}\theta_\ell}\right)}. \tag{3.11}$$

2. *Every spike generated by the Gaussian $\mathcal{G}_{\mu,t_{\ell-1,k}}$ lies in the interval $(t_{\ell-1,k} - \zeta, t_{\ell-1,k} + \zeta)$.*
3. *For each $k \in [\![K]\!]$, denote by $N_{\max}(k)$ the maximal number of spikes in $\mathcal{T}_{\ell,p}$ generated around the Gaussian $\mathcal{G}_{\mu,t_{\ell-1,k}}$. Then*

$$N_{\max}(k) \leq \left\lfloor 1 + \log_2\left(\frac{\omega_{\ell,p}}{\tau_{\ell,p}\theta_{\ell,p}}\right)\right\rfloor,$$

and the total number of spike times in $\mathcal{T}_{\ell,p}$ satisfies

$$K \leq |\mathcal{T}_{\ell,p}| \leq K \left\lceil 1 + \log_2 \left(\frac{\omega_{\ell,p}}{\tau_{\ell,p}\theta_{\ell,p}} \right) \right\rceil. \tag{3.12}$$

In particular, if $\omega_{\ell,p} < 2\tau_{\ell,p}\theta_{\ell,p}$, we have $|\mathcal{T}_{\ell,p}| = K$.

Proof. First of all, because of the separation condition (3.9), we can isolate and analyze separately the dynamics around each spike time $t_{\ell-1,k} \in \mathbb{T}_{\ell-1}$. Indeed, for all $k \in [\![K]\!]$, we have that

$$J_\ell(t) \sim \mathcal{G}_{\mu,t_{\ell-1,k}}(t),$$

since $\mathcal{G}_{\mu,t_{\ell-1,k}}$ is negligible in the interval $[t_{\ell-1,j} - 3\mu, t_{\ell-1,j} + 3\mu]$ with $j \neq k$.

Moreover, the amplitude condition (3.10) ensures that the equation $\omega_{\ell,p}\mathcal{G}_{\mu,t_{\ell-1,k}}(t) = \theta_{\ell,p}$ has two solutions

$$t_{\ell,p,k}^- = t_{\ell-1,k} - \zeta \quad \text{and} \quad t_{\ell,p,k}^+ = t_{\ell-1,k} + \zeta,$$

with ζ as in (3.11). By continuity, the ODE solution $\xi_{\ell,p}(t)$ must reach the threshold for some

$$t_{\ell,k,1} \in (t_{\ell-1,k} - \zeta, t_{\ell-1,k}).$$

Now, because $t_{\ell,k,1}$ occurs on the rising side of the Gaussian $\mathcal{G}_{\mu,t_{\ell-1,k}}(t)$, at least half of the Gaussian's total area remains to the right of $t_{\ell,k,1}$. Hence the remaining effective charge after the first reset is at most

$$Q_{k,1} \leq \frac{1}{2} \frac{\omega_{\ell,p}}{\tau_{\ell,p}} \int_{\mathbb{R}} \mathcal{G}_{\mu,t_{\ell-1,k}}(t)\, dt = \frac{\omega_{\ell,p}}{2\tau_{\ell,p}}.$$

A second spike requires that this residual charge can raise again $\xi_{\ell,p}$ from 0 to $\theta_{\ell,p}$. In particular, we need

$$\frac{\omega_{\ell,p}}{2\tau_{\ell,p}} \geq \theta_{\ell,p}$$

or, equivalently, $\omega_{\ell,p} \geq 2\tau_{\ell,p}\theta_{\ell,p}$.

Suppose this happens and a second spike occurs at $t_{\ell,k,2} \in (t_{\ell-1,k}, t_{\ell-1,k} + \zeta)$. By then, at most half of the remaining bump area has been consumed. So the residual charge after the second reset satisfies

$$Q_{k,2} \leq \frac{\omega_{\ell,p}}{4\tau_{\ell,p}}.$$

Thus a third spike requires $\omega_{\ell,p} \geq 4\tau_{\ell,p}\theta_{\ell,p}$. In general, after n spikes, the remaining charge is bounded by

$$Q_{k,n} \leq \frac{\omega_{\ell,p}}{2^n \tau_{\ell,p}},$$

and an $(n+1)$-th spike requires $\omega_{\ell,p} \geq 2^n \tau_{\ell,p}\theta_{\ell,p}$. Thus the maximal number $N_{\max}(k)$ of spikes in $\mathcal{T}_{\ell,p}$ around $t_{\ell-1,k} \in \mathbb{T}_{\ell-1}$ satisfies

$$\frac{\omega_{\ell,p}}{\tau_{\ell,p}\theta_{\ell,p}} \geq 2^{N_{\max}(k)},$$

i.e.

$$N_{\max} \leq \left\lceil 1 + \log_2 \left(\frac{\omega_{\ell,p}}{\tau_{\ell,p}\theta_{\ell,p}} \right) \right\rceil, \quad \text{for all } k \in [\![K]\!].$$

Finally, since the Gaussian bumps are independent (by separation hypothesis (3.9)), we can sum over k the contributions to obtain the final estimate (3.12) for the number of spikes $|\mathcal{T}_{\ell,p}|$. $\qquad\square$

11

Figure 3.1 illustrates Proposition 3.4 in the case $P = 1$. We fix $\theta_\ell = 0.2$, choose $\mu = 0.8$, and prescribe presynaptic spike times $\mathbb{T}_{\ell-1} = \{20, 35, 50\}$, which satisfy the separation condition (3.9). In the top panel, we set $\tau_\ell = 5$ and $\omega_\ell = 1.5$, so that $\omega_\ell/(\tau_\ell\theta_\ell) = 1.5 < 2$. In this case, the postsynaptic neuron generates exactly three spikes, located at $\mathbb{T}_\ell = \{19.5, 34.5, 49.5\}$. In the bottom panel, instead, we take $\tau_\ell = 5$ and $\omega_\ell = 3$, yielding $\omega_\ell/(\tau_\ell\theta_\ell) = 3 > 2$. Here, the same presynaptic input produces six postsynaptic spikes, namely $\mathbb{T}_\ell = \{19.6, 20.5, 34.7, 35.5, 49.7, 50.5\}$.
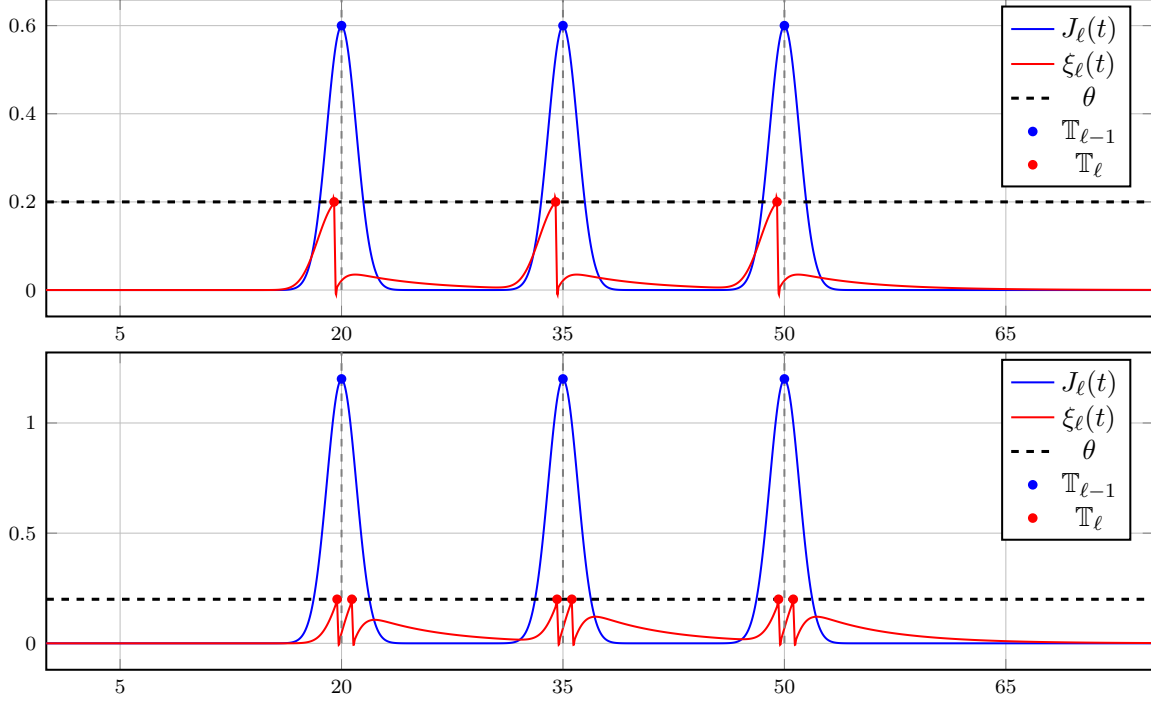

FIGURE 3.1. Graphical illustration of Proposition 3.4. The plots show how the number of postsynaptic spikes depends on the ratio $\omega_{\ell,p}/(\tau_{\ell,p}\theta_{\ell,p})$. In the top panel, when this ratio is below the critical threshold 2, each presynaptic spike produces exactly one postsynaptic spike. In the bottom panel, when the ratio exceeds 2, each presynaptic spike gives rise to two postsynaptic spikes. This demonstrates how the parameters $\tau_{\ell,p}$, $\omega_{\ell,p}$, and $\theta_{\ell,p}$ govern the propagation and multiplication of spike times across layers.

Proposition 3.5 (Spike behavior under overlapping input bumps). *Fix $\ell \in [\![L]\!]$, $p \in [\![P]\!]$, and let $\xi_{\ell,p}(t)$ be the membrane potential of a hidden-layer neuron governed by the hybrid system (Hidd), with input current $J_\ell(t)$ given by (2.1) and input spike times $\mathbb{T}_{\ell-1} = \{t_{\ell-1,k}\}_{k=1}^K \subset (0,T)$ that may be arbitrarily close. Let*

$$\mathcal{M} := \left\{ t^* \in (0,T) \ : \ J_\ell'(t^*) = 0 \text{ and } J_\ell''(t^*) < 0 \right\}$$

be the set of strict local maxima of J_ℓ and suppose that

$$\omega_{\ell,p} \max_{t \in \mathcal{M}} J_\ell(t) > \theta_{\ell,p}.$$

Then, it holds the following

   1. *Each maximum $t^* \in \mathcal{M}$ generates at least one spike in a neighborhood of t^*. In particular, $|\mathcal{T}_{\ell,p}| \geq |\mathcal{M}|$.*

12

2. *For each maximum $t^* \in \mathcal{M}$, the number of spikes it can generate is bounded by*

$$N_{\max}(t^*) \leq \left\lfloor 1 + \log_2\left(\frac{\omega_{\ell,p}}{\tau_{\ell,p}\theta_{\ell,p}}\right)\right\rfloor.$$

Hence

$$|\mathcal{T}_{\ell,p}| \leq |\mathcal{M}| \left\lfloor 1 + \log_2\left(\frac{\omega_{\ell,p}}{\tau_{\ell,p}\theta_{\ell,p}}\right)\right\rfloor. \tag{3.13}$$

In particular, if $\omega_{\ell,p} < 2\tau_{\ell,p}\theta_{\ell,p}$, we have $|\mathcal{T}_{\ell,p}| = |\mathcal{M}|$.

Proof. Let $t^* \in \mathcal{M}$ be a strict local maximum of $J_\ell(t)$. Since $\xi_{\ell,p}$ is continuous and initially small, and $\omega_{\ell,p}J_\ell(t^*) > \theta_{\ell,p}$ by assumption, the ODE ensures that in a neighborhood of t^*, $\xi_{\ell,p}$ grows above the threshold. Thus there exists a first threshold crossing time $t_{\ell,p,1}$ near t^*. Therefore, each $t^* \in \mathcal{M}$ yields at least one spike and we have $|\mathcal{T}_{\ell,p}| \geq |\mathcal{M}|$.

To obtain the upper bound (3.13), we adopt the same argument as in the proof of Proposition 3.4. At each $t^* \in \mathcal{M}$, $J_\ell(t)$ behaves like a wide unimodal bump (a sum of overlapping Gaussians). The total effective charge available from this bump is

$$Q = \frac{\omega_{\ell,p}}{\tau_{\ell,p}}\int_{\mathbb{R}} J_\ell(s)\,ds.$$

Now, each spike consumes at least $\theta_{\ell,p}$ units of charge raising $\xi_{\ell,p}$ from 0 to the threshold $\theta_{\ell,p}$. Because the bump is unimodal, after each threshold crossing, at least half the bump's residual area lies ahead of the crossing time. Thus the available charge after n spikes is bounded by $Q2^{-n}$. So to produce an $(n+1)$-th spike, we need $Q2^{-n} \geq \theta_{\ell,p}$, which is equivalent to $\omega_{\ell,p} \geq 2^n\tau_{\ell,p}\theta_{\ell,p}$. Therefore the maximum number of spikes satisfies (3.13). $\square$

Propositions 3.4 and 3.5 show that, under natural separation or unimodal bump conditions, the number of spikes per layer typically forms a non-increasing sequence. In particular, when presynaptic spike times are sufficiently separated and the ratio

$$\Gamma_{\ell,p} := \frac{\omega_{\ell,p}}{\tau_{\ell,p}\theta_{\ell,p}} < 2,$$

each input bump produces at most one output spike, leading to stability of spike counts across layers.

This quantity $\Gamma_{\ell,p}$ measures the effective input strength per unit time relative to the firing threshold $\theta_{\ell,p}$. Intuitively, if $\Gamma_{\ell,p} < 2$, each bump is too weak to cause more than one firing. On the other hand, if $\Gamma_{\ell,p} \geq 2$, residual charge after a spike may suffice to trigger additional firings. In this sense, we say the neuron operates in a **high-gain regime** when $\Gamma_{\ell,p} \gg 1$.

Furthermore, the membrane time constant $\tau_{\ell,p}$ controls how quickly the potential leaks away. Here, we can identify two regimes

1. **long memory** (large $\tau_{\ell,p}$) means the neuron retains input longer, making extra spikes more likely;
2. **short memory** (small $\tau_{\ell,p}$) means quick decay, which usually suppresses multiple firings unless bumps overlap.

Under these notions, the exceptions to monotonicity in the number of spikes per neuron and layer occur in two main scenarios:

1. **Overlapping presynaptic spikes.** If the intervals between spike times in $\mathbb{T}_{\ell-1}$ violate the separation condition (3.9), Gaussian bumps overlap to form a composite input $J_\ell(t)$ with several local maxima. As Proposition 3.5 indicates, each strict local maximum may induce at least one spike, so that $|\mathcal{T}_{\ell,p}| \geq |\mathcal{M}|$, where $|\mathcal{M}|$ is the number of maxima of J_ℓ.

2. **High gain with sufficient memory.** If the ratio $\Gamma_{\ell,p}$ satisfies $\Gamma_{\ell,p} \geq 2$, then, as in Proposition 3.4, a single input bump may yield multiple spikes. Each reset leaves enough residual input charge for another threshold crossing. The maximal number of spikes per bump grows logarithmically with $\Gamma_{\ell,p}$.

When both overlap and high gain occur simultaneously, the increase in spike count can be particularly pronounced, as multiple overlapping bumps may each sustain several spikes after resets. Overall, we have the following possibilities

$$|\mathcal{T}_{\ell,p}| = |\mathbb{T}_{\ell-1}|, \quad \text{if inputs are separated and } \Gamma_{\ell,p} < 2,$$

$$|\mathcal{T}_{\ell,p}| \leq |\mathbb{T}_{\ell-1}|, \quad \text{if inputs are separated but } \Gamma_{\ell,p} \geq 2,$$

$$|\mathcal{T}_{\ell,p}| \geq |\mathcal{M}|, \quad\;\; \text{if overlapping inputs create } |\mathcal{M}| \text{ local maxima,}$$

$$|\mathcal{T}_{\ell,p}| > |\mathbb{T}_{\ell-1}|, \quad \text{if overlapping inputs and } \Gamma_{\ell,p} \geq 2.$$

In summary, while the generic trend is non-increasing spike counts across layers, increases may occur under specific conditions: overlapping inputs, sufficiently large gain-to-threshold ratios, and long enough membrane time constants to permit resets between peaks. These situations are exceptional and often unstable, but they provide a precise mechanism for how the number of spike times among layers may grow.

Finally, recall that at the level of the full layer, recall that the set of spike times is given by $\mathbb{T}_\ell = \bigcup_{p=1}^{P} \mathcal{T}_{\ell,p}$. Since Proposition 3.4 characterizes the number of spikes neuron-wise, the total number of spikes in the layer is obtained by summing over all P neurons. In particular, the bounds in (3.12) scale linearly with P. This observation highlights the role of width: layers with few neurons naturally limit the overall growth of spike counts, whereas wide layers may amplify the total number of spikes, even if each individual neuron remains in a controlled regime. From a biological perspective, this scaling resonates with the principle of metabolic efficiency, since neural systems often rely on sparse populations of neurons to encode information in an energy-efficient way [17].

Remark 3.6. According to what we have observed so far, due to the reset dynamics (Hidd), in general each neuron emits at most one spike per sufficiently separated input bump. Hence, if the presynaptic spikes are non-overlapping and well-behaved, the number of output spikes per neuron cannot exceed the number of input spikes, meaning that the sequence of spikes number is non-increasing under non-pathological conditions.

This means that, if we exclude pathological conditions such as resonance effects, overlapping spikes inducing multiple firings per bump, or unusual parameter regimes (tiny thresholds, long memory time constants), then the number $|\mathbb{T}_0|$ of spike points in the input layer allows defining an upper bound on the number of spike points in the deeper layers.

Finally, recall that he input layer has a single neuron, governed by the ODE (In), and that, according to Proposition 3.1, the set of its spike times $\mathbb{T}_0 \subset (0, T)$ is

- Empty, if $\langle \boldsymbol{a}, \boldsymbol{x} \rangle \leq \theta_v$ or if the time horizon T is not large enough so to allow the neuron generate a spike;
- A periodically spaced set $\{k\beta\}_{k=1}^{K}$, otherwise.

Hence, the number of spikes in the input layer is determined entirely by the input $\boldsymbol{x}$ and by the tuning of the parameter $\boldsymbol{a}$ modulating how the input neuron $\mathcal{N}^{\text{in}}$ processes $\boldsymbol{x}$.

4. UNIVERSAL APPROXIMATION

In this section we investigate the expressive power of the SNN architecture introduced in Section 2. Our goal is to show that the class of mappings SNN : $\mathbb{R}^d \to \mathbb{R}$ defined in (2.5), obtained by combining the LIF dynamics in the input, hidden, and output layers with the static Cybenko-type readout, is dense in the space of continuous functions on compact subsets of $\mathbb{R}^d$. This result provides

a rigorous justification for the use of the proposed architecture: despite the hybrid dynamical nature of the underlying models, the inclusion of the readout layer ensures that the network retains the universal approximation property established for classical feedforward neural networks [5, 19].

Theorem 4.1. *Let ρ be a Borel probability measure with compact support $\Omega \Subset \mathbb{R}^d$, and let $f \in C_0(\Omega) \cap L^2(\rho)$. Let $\sigma : \mathbb{R} \to \mathbb{R}$ be a Lipschitz, non-polynomial activation function. Then, for every $\varepsilon > 0$, there exists $L, P \in \mathbb{N}^*$, $\mu > 0$, and a Spiking Neural Network $\mathrm{SNN} : \mathbb{R}^d \to \mathbb{R}$ with architecture given by* (2.5) *such that*

$$\|\mathrm{SNN} - f\|_{L^2(\rho)} < \varepsilon,$$

provided

$$\mu \leq \frac{\varepsilon}{C},$$

where $C = C(\sigma, T, w, \tau_u, \boldsymbol{\nu}) > 0$ depends only on the regularity of σ, the time interval T, the parameters w and τ_u in (Out)*, and the readout parameters $\boldsymbol{\nu} = \{\nu_p\}_{p=1}^P$.*

Proof. Before entering the proof, let us highlight that our SNN architecture is rigorously defined in Subsection 2.1.5 via the mollified dynamics (Out), which guarantee well-posedness and trainability. Nevertheless, for the constructive part of the universal approximation argument, it will be convenient to temporarily work with the idealized δ-driven system (2.4). As explained in Remark 2.1, although this system is not well-posed in the classical sense, one may still assign a natural formal solution by superposing exponential responses to delta inputs. This explicit representation provides a convenient proxy that allows us to design spike trains encoding prescribed target values. Its role is justified by the fact that the mollified dynamics (Out) approximate the δ-driven ones as $\mu \to 0^+$, ensuring that the final approximation result holds for the actual SNN architecture.

We now divide the proof into multiple steps.

Step 1: approximation by Shallow Neural Networks. By [19, Theorem 1], since $f \in C_0(\Omega) \cap L^2(\rho)$ and σ is non-polynomial, there exists an integer $P \in \mathbb{N}^*$, weights $\{\nu_p\}_{p=1}^P \subset \mathbb{R}$, and directions $\{\boldsymbol{\alpha}_p\}_{p=1}^P \subset \Omega$ such that for all $\boldsymbol{x} \in \Omega$ the function

$$f_P(\boldsymbol{x}) = \sum_{p=1}^P \nu_p \sigma(\langle \boldsymbol{\alpha}_p, \boldsymbol{x} \rangle)$$

satisfies

$$\|f - f_P\|_{L^2(\rho)} < \frac{\varepsilon}{2}.$$

Step 2: Construction of SNN subnets to approximate basis units. For each $p \in [\![P]\!]$ and $K \in \mathbb{N}^*$ fixed, we want to design spike times $\mathcal{T}_{L,p} = \{t_{L,p,1}, \ldots, t_{L,p,K}\} \subset [0, T]$ such that the corresponding solution $u_{p,\delta}$ of (2.4) satisfies

$$u_{p,\delta}(T) = \theta_u + \langle \boldsymbol{\alpha}_p, \boldsymbol{x} \rangle, \tag{4.1}$$

with $\{\boldsymbol{\alpha}_p\}_{p=1}^P$ as in Step 1. To this end, let us first recall (see Remark 2.1) that this final-time solution is formally given by

$$u_{p,\delta}(T) = \frac{w}{\tau_u} \sum_{k=1}^K e^{-\frac{T - t_{L,p,k}}{\tau_u}}.$$

15

Now, define the function

$$S : [0, T]^K \to \mathbb{R}, \quad S(t_{L,p,1}, \ldots, t_{L,p,K}) := \sum_{k=1}^{K} e^{-\frac{T - t_{L,p,k}}{\tau_u}},$$

and notice that S is continuous $[0, T]^K$ since each exponential term is continuous in $t_{L,p,k}$, and sums of continuous functions are continuous. Moreover, S is strictly increasing in each coordinate $t_{L,p,k}$ and

$$\text{if all } t_{L,p,k} = 0, \text{ then } S = K e^{-\frac{T}{\tau_u}},$$
$$\text{if all } t_{L,p,k} = T, \text{ then } S = K.$$

Hence, S is surjective onto its range

$$S([0, T]^K) = \left[K e^{-\frac{T}{\tau_u}}, K \right].$$

Let now

$$S^* := \frac{\tau_u}{w} \left(\theta_u + \langle \boldsymbol{\alpha}_j, \boldsymbol{x} \rangle \right),$$

and assume

$$S^* \in \left[K e^{-\frac{T}{\tau_u}}, K \right]. \tag{4.2}$$

Since S is continuous and maps a compact set onto this interval, the intermediate value theorem ensures that there exists a tuple $(t_{L,p,1}, \ldots, t_{L,p,K}) \in [0, T]^K$ such that $S(t_{L,p,1}, \ldots, t_{L,p,K}) = S^*$. In conclusion, we can always choose spike times to fulfill (4.1), as long as K is large enough so that (4.2) holds.

Step 3: error estimates. For all $p \in [\![P]\!]$, let $\mathcal{T}_{L,p}$ be constructed as in Step 2, and let $u_p(T)$, $u_{p,\delta}(T)$ be the corresponding final-time solutions of (Out) and (2.4). Moreover, let

$$\Delta u^{(p)} := u_p(T) - u_{p,\delta}(T) = \frac{w}{\tau_u} e^{-\frac{T}{\tau_u}} \sum_{t^* \in \mathcal{T}_{L,p}} \left[\int_0^T e^{\frac{s}{\tau_u}} \mathcal{G}_{\mu,t^*}(s) \, ds - e^{\frac{t^*}{\tau_u}} \right]$$

$$= \frac{w}{\tau_u} e^{-\frac{T}{\tau_u}} \sum_{t^* \in \mathcal{T}_{L,p}} \left[\int_0^T \varphi(s) \mathcal{G}_{\mu,t^*}(s) \, ds - \varphi(t^*) \right],$$

with $\varphi(s) := e^{\tau_u^{-1} s}$. Then, since $\mathcal{G}_{\mu,t^*}(\cdot)$ is a Gaussian (in particular, a mollifier), we easily get that

$$\left| \int_0^T \varphi(s) \mathcal{G}_{\mu,t^*}(s) ds - \varphi(t^*) \right| \leq \mu \sup_{s \in [0,T]} |\varphi'(s)| = \frac{\mu}{\tau_u} e^{\frac{T}{\tau_u}}.$$

Using this and the definition of $\mathcal{T}_{L,p}$, we can then estimate

$$|\Delta u^{(p)}| \leq \frac{\mu |w|}{\tau_u^2} |\mathcal{T}_{L,p}| = \frac{K \mu |w|}{\tau_u^2}.$$

Hence, if we define $z_p := \sigma(u_p(T) - \theta_u)$, using (4.1) and the above estimate we get

$$\left| z_p - \sigma(\langle \boldsymbol{\alpha}_p, \boldsymbol{x} \rangle) \right| = \left| \sigma(u_p(T) - \theta_u) - \sigma(u_{p,\delta}(T) - \theta_u) \right| \leq \mathrm{Lip}(\sigma) |\Delta u^{(p)}| \leq \frac{K \mu |w| \mathrm{Lip}(\sigma)}{\tau_u^2},$$

16

where $\text{Lip}(\sigma)$ denotes the Lipschitz constant of σ. Using this, we can estimate the total error

$$\|\text{SNN} - f\|_{L^2(\rho)} \leq \|\text{SNN} - f_P\|_{L^2(\rho)} + \|f_P - f\|_{L^2(\rho)}$$

$$< \sum_{p=1}^{P} |\nu_p| \cdot \|\sigma(u_p(T) - \theta_u) - \sigma(\langle \boldsymbol{\alpha}_p, \boldsymbol{x} \rangle)\|_{L^2(\rho)} + \frac{\varepsilon}{2}$$

$$\leq \frac{K\mu|w|\text{Lip}(\sigma)}{\tau_u^2} \sum_{p=1}^{P} |\nu_p| + \frac{\varepsilon}{2} \leq \frac{K\mu|w|\|\boldsymbol{\nu}\|_1 \text{Lip}(\sigma)}{\tau_u^2} + \frac{\varepsilon}{2}.$$

Hence, if

$$\mu \leq \frac{\varepsilon \tau_u^2}{2K|w|\|\boldsymbol{\nu}\|_1 \text{Lip}(\sigma)},$$

we finally have $\|\text{SNN} - f\|_{L^2(\rho)} < \varepsilon$. □

4.1. On the necessity of delta and Gaussian dynamics in the universal approximation argument.

The proof of Theorem 4.1 relies crucially on the interplay between two different formulations of the spiking neuron dynamics: the idealized model (2.4) involving Dirac delta functions, and the regularized model (Out) based on smooth Gaussian kernels. Each of these plays an essential but distinct role in establishing the universal approximation property, and neither alone would suffice.

Concretely, the δ-based model (2.4) is employed as a mathematical abstraction to construct spike trains that precisely encode the values of the target function to be approximated. At this regards, we shall highlight that, although spike times are generated by the intrinsic dynamics of the LIF model, our constructive arguments show that, by suitable parameter choices, one can ensure that the resulting dynamics produce spike trains with prescribed timings.

In this sense, the spike times can be designed indirectly. In our case, because the response of $u_{p,\delta}(T)$ to a spike at time t^* is an exponential term

$$e^{-\frac{T-t^*}{\tau_u}},$$

the final value becomes an explicit sum:

$$u_{p,\delta}(T) = \frac{w}{\tau_u} \sum_{k=1}^{K} e^{-\frac{T-t_{L,p,k}}{\tau_u}}.$$

This formula allows for direct control of the output by carefully choosing the spike times $\{t_{L,p,k}\}$, since each spike contributes independently and additively. Thus, one can solve an inverse problem – finding spike times that realize a prescribed value – using a well-understood basis of exponential functions.

This tractability fails in the Gaussian-regularized model, where the dynamics are instead governed by a model (Out) involving smooth Gaussian bumps centered at the spike times. In this setting, the solution $u_p(T)$ is no longer a clean sum of isolated terms – each spike's contribution spreads over time, and neighboring spikes interfere with one another due to the overlap of Gaussians and the integration over the full interval. As a result, the map from spike times to $u_p(T)$ becomes nonlinear, entangled, and difficult to invert explicitly, especially as the number of spikes grows. Therefore, the Gaussian model does not allow a direct constructive encoding of target values.

On the other hand, the Gaussian-regularized model is essential for defining the actual network implementation, as it yields smooth, well-posed dynamics that are amenable to both mathematical analysis and numerical simulation. In contrast, the δ-based model involves impulsive right-hand

sides with Dirac distributions, which make the differential equations ill-posed in the classical sense. Specifically:

1. The equation does not admit solutions in standard function spaces like C^1 or L^2, but only in the sense of distributions or measures.
2. The membrane potential $u_{p,\delta}(t)$ becomes discontinuous or non-differentiable at spike times, which complicates the definition and uniqueness of solutions.

By replacing each delta function $\delta(t - t^*)$ with a smooth, compactly supported approximation $G_{\mu,t^*}(t)$, we recover a system with continuous and differentiable dynamics that admits a unique classical solution and can be treated using standard ODE theory.

In particular, the membrane potential $u_p(t)$ becomes a smooth function of time, even in the presence of multiple spikes, and the dependence of $u_p(T)$ on spike times and network parameters becomes differentiable, which is crucial for computing gradients and training the network via standard optimization techniques.

Therefore, the Gaussian model serves as a practically and mathematically viable surrogate for the ideal δ-based model. While it does not allow for explicit spike-time design, its smoothness enables trainability and implementability, and we recover expressiveness by showing that its behavior converges to that of the delta model as the smoothing parameter $\mu \to 0^+$.

In conclusion, both systems are indispensable: the δ-based model provides a constructive encoding mechanism for function values via spike timing, while the Gaussian-based model ensures smooth realizability and robust approximation of the desired behavior.

This dual modeling strategy mirrors classical approximation theory, where one constructs an exact interpolant in an idealized basis (e.g., splines, wavelets) and then implements an approximate version via practical basis functions [7, 12, 23]. In our setting, spikes and their temporal summation play the role of such basis elements, and Gaussian smoothing ensures continuity and differentiability without sacrificing expressive power.

4.2. On the role of hidden layers and the depth–width tradeoff.
The proof of Theorem 4.1 exploits only the explicit representation of the output potentials $u_p(T)$, without relying on the detailed dynamics of the input and hidden layers. One might therefore wonder whether the hidden layers are dispensable, and whether an architecture consisting solely of the output layer (Out) would already suffice for universal approximation.

From a purely functional-analytic standpoint, this is correct: universality follows from the ability of the output neurons to generate sums of exponential responses to spike times, which in principle can be prescribed directly. In this abstract sense, hidden layers do not affect the *existence* of an approximating network. Nevertheless, the hidden layers remain indispensable for several reasons:

1. **Encoding mechanism.** The output layer does not receive the input $\boldsymbol{x}$ directly: it only integrates spike trains. Without hidden layers, there is no mechanism to transform the input into spike patterns that the output neurons can process. In other words, the hidden layers are the only part of the architecture that connects the static input domain to the temporal structure required by the output layer. This role cannot be bypassed without changing the model itself.
2. **Information richness and scalability.** The sets of spike trains $\mathcal{T}_{\ell,p}$ generated in the hidden layers' neurons form the effective dictionary from which the output layer builds its exponential sums. The number of layers L and neurons per layer P directly determine how large and flexible this dictionary is. A shallow or narrow hidden representation may only generate a limited repertoire of spike trains, leading to poor expressive capacity in practice even if universality holds in principle.
3. **Minimal architectures and analogy with neural ODEs.** In analogy with the classical universal approximation theorems, one hidden layer with sufficiently many neurons is

expected to suffice for universality. Conversely, one may drastically reduce the width if depth is increased: with only one neuron per layer but sufficiently many layers, universality should still hold. This mirrors the results of [1], where the authors prove that in ResNet-type architectures interpolation can be achieved either with a single wide layer or with many narrow layers. While our proof does not establish such a precise depth–width tradeoff for SNNs, it is consistent with this expectation and suggests that a similar phenomenon underlies the universality of spiking architectures.

In summary, although the formal proof of Theorem 4.1 is phrased entirely in terms of the output layer, the hidden layers are structurally necessary to link the static input $\boldsymbol{x}$ with the temporal spike-based encoding on which the output relies. Their number and size control the richness of the available encodings, and therefore the efficiency and robustness of approximation. In this sense, the hidden layers are not an optional embellishment, but the essential mechanism through which SNNs realize universal approximation.

To see this more concretely, note that if one were to remove all hidden layers and keep only the output dynamics, then the spike times feeding the output neurons would have to be fixed a priori, say $\mathcal{T} = \{t_1^*, \ldots, t_m^*\}$, with no dependence on the input $\boldsymbol{x}$. The outputs would then be only dependent on the time horizon T. Such a device can only approximate constant functions on the input space, and in that sense it is not a neural network at all, since it lacks an input–output relation.

By contrast, with even a single hidden layer the spike times $\mathcal{T}_{1,p}(\boldsymbol{x})$ are generated dynamically in response to $\boldsymbol{x}$. The output then becomes dependent on $\boldsymbol{x}$ through the spike locations. This input dependence is exactly what gives SNNs their expressive power. In this sense, the hidden layers are therefore not cosmetic additions but the indispensable mechanism that transforms the output dynamics into a genuine SNN capable of universal approximation.

4.3. **Extension to multiclass classification.** Although in our analysis we have restricted ourselves to the case of binary classification, where the dataset labels are of the form $y_i \in \{0, 1\}$, the framework extends naturally to an arbitrary number of classes without altering the dynamical analysis or the universal approximation result. This requires a simple modification, which consists in replacing the single readout neuron $\mathcal{N}^{\text{read}}$ by a layer of $C \geq 2$ readout neurons $\{\mathcal{N}_c^{\text{read}}\}_{c=1}^C$, each producing a final-time potential

$$\mathcal{R}^{(c)}\big(\boldsymbol{u}(T, \boldsymbol{x}_i)\big), \quad c \in [\![C]\!],$$

for each datum $\boldsymbol{x}_i$, and generating the predictions via the softmax mapping:

$$y_i^{(c)} = \mathcal{R}^{(c)}\big(\boldsymbol{u}(T, \boldsymbol{x}_i)\big) \left(\sum_{j=1}^C \mathcal{R}^{(j)}(\boldsymbol{u}(T, \boldsymbol{x}_i))\right)^{-1}, \quad c \in [\![C]\!].$$

Notice that the well-posedness results of Section 3, which establish the existence, uniqueness, and characterization of spike times, remain unchanged. They are proved neuron-wise and are independent of the number of readout neurons.

Moreover, the Universal Approximation Theorem 4.1, stated for scalar-valued target functions, extends immediately to vector-valued mappings $f : \Omega \to \mathbb{R}^C$ by applying the construction component-wise.

In conclusion, the transition from binary to multiclass classification requires only a modification of the final readout layer, and all the analytical results on well-posedness and universal approximation remain valid without major alteration.

5. Learning framework

Denote

$$\Upsilon := \Big(\boldsymbol{a}, (\boldsymbol{\omega}_1, \ldots, \boldsymbol{\omega}_L), w \Big) \in \mathbb{R}^d \times \mathbb{R}^{PL} \times \mathbb{R}$$

$$\boldsymbol{\nu} = (\nu_1, \ldots, \nu_P) \in \mathbb{R}^P \tag{5.1}$$

the parameters of our SNN, with Υ indicating the "dynamics" parameters of the input, hidden and output layers, and ν the parameters of the static readout layer. Their training will be carried out through standard empirical risk minimization, i.e. minimizing the cost functional

$$G(\Upsilon, \boldsymbol{\nu}) = \frac{1}{N} \sum_{i=1}^{N} \mathrm{loss}\Big(\mathcal{R}(\boldsymbol{u}(T, \boldsymbol{x}_i), y_i \Big) + \gamma \Big(\|\Upsilon\|^2 + \|\nu\|^2 \Big), \tag{5.2}$$

where $\gamma > 0$ is a regularization parameter, $\sigma : \mathbb{R} \to (0,1)$ is a Lipschitz activation function, $\mathrm{loss} : \mathbb{R} \to \mathbb{R}^+$ is a suitable loss function, and

$$\|\Upsilon\|^2 = \|\boldsymbol{a}\|^2 + \sum_{\ell=1}^{L} \|\boldsymbol{\omega}_\ell\|^2 + |w|^2.$$

To minimize (5.2) we will apply a gradient-based algorithm, which requires computing the gradients $\nabla_\Upsilon G$ and $\nabla_{\boldsymbol{\nu}} G$ with respect to the model parameters.

For what concerns the readout parameters $\boldsymbol{\nu}$, this gradient is obtained as a direct application of the chain rules for derivatives. In particular, if we denote by $z_i = \mathcal{R}(\boldsymbol{u}(T, \boldsymbol{x}_i))$ the network's prediction on the i-th datum $\boldsymbol{x}_i$, we have that

$$\nabla_{\nu_p} G = 2\gamma \nu_p + \frac{1}{N} \sum_{i=1}^{N} \partial_{z_i} \mathrm{loss}(z_i, y_i) \sigma(u_p(T) - \theta_u), \quad \text{for all } p \in [\![P]\!].$$

As for $\nabla_\Upsilon G$, the computation is more delicate due to the reset mechanisms in the input and hidden layers, which introduces non-differentiability points and jumps in the state trajectories of the LIF dynamics. To rigorously address this issue, we will apply a "surrogate gradient" approach, based on the mollification of the discontinuous spike events by replacing the hard thresholding in the LIF model with differentiable surrogate functions (see, e.g., [8]). This yields smooth approximations of the state trajectories and well-defined gradients $\nabla_\Upsilon G$ while preserving the qualitative spiking behavior. Once the dynamics are regularized, the computation of $\nabla_\Upsilon G$ can be rigorously formulated through an optimal control framework. The resulting first-order optimality system consists of a coupled forward–backward problem: the forward dynamics describe the evolution of membrane potentials and synaptic currents, while the adjoint dynamics propagate the sensitivity of the objective functional with respect to state perturbations. This adjoint methodology provides a principled way to evaluate parameter gradients, in direct analogy with Pontryagin's Maximum Principle [27, 28, 30] or backpropagation through time [18].

5.1. Computation of $\nabla_\Upsilon G$. Let

$$\mathbb{T} := \bigcup_{\ell=0}^{L} \mathbb{T}_\ell$$

be the set of all spike times of the input and hidden layers. For all $t \in (0, T)$ and parameters Υ as in (5.1), we define the stacked state vector

$$X(t) = \Big(v(t), \{\boldsymbol{\xi}_\ell(t)\}_{\ell=1}^{L}, \boldsymbol{u}(t) \Big)^\top = \Big(v(t), \big\{ (\xi_{\ell,1}(t), \ldots, \xi_{\ell,P}(t)) \big\}_{\ell=1}^{L}, \big(u_1(t), \ldots, u_P(t) \big) \Big)^\top$$

and the stacked dynamics

$$\mathcal{F}(X(t), \Upsilon) = \left(f_v, \{f_{\ell,1}, \ldots, f_{\ell,P}\}_{p=1}^P, \{f_{u_p}\}_{p=1}^P\right)^\top,$$

where

$$
\begin{aligned}
\text{Input layer} &\quad f_v := \tau_v^{-1}\left(-v + \langle \boldsymbol{a}, \boldsymbol{x}_i \rangle\right) &\quad \text{for all } i \in [\![N]\!] \\
\text{Hidden layers} &\quad f_{\xi_{\ell,p}} := \tau_{\ell,p}^{-1}\left(-\xi_{\ell,p} + \omega_{\ell,p} J_\ell\right) &\quad \text{for all } \ell \in [\![L]\!] \text{ and } p \in [\![P]\!] \\
\text{Output layer} &\quad f_{u_p} := \tau_u^{-1}\left(-u_p + w\Phi_p\right) &\quad \text{for all } p \in [\![P]\!]
\end{aligned}
\tag{5.3}
$$

With this notations, the input-hidden-output layer evolution of our SNN is governed by the ODE system

$$
\begin{cases}
\dot{X}(t) = \mathcal{F}(X(t), \Upsilon), \quad t \in (0, T) \setminus \mathbb{T} \\
X(0) = 0 \\
X^+ = \mathcal{J}(X^-)
\end{cases}
\tag{5.4}
$$

where $X^+ = \mathcal{J}(X^-)$ indicates the jump conditions

$$v_k^- = \theta_v, \ v_k^+ = 0 \quad \text{and} \quad \xi_{\ell,p,k}^- = \theta_{\ell,p}, \ \xi_{\ell,p,k}^+ = 0.$$

Moreover, by applying standard variational theory, we know that

$$\nabla_\Upsilon G = 2\gamma\Upsilon - \int_0^T \lambda(t)^\top \partial_\Upsilon \mathcal{F}\left(X(t), \Upsilon\right) dt,$$

where the adjoint states

$$\lambda(t) = \left(\lambda_v(t), \{\lambda_{\boldsymbol{\xi}_\ell}(t)\}_{\ell=1}^L, \lambda_{\boldsymbol{u}(t)}\right) = \left(\lambda_v(t), \{(\lambda_{\xi_{\ell,1}}(t), \ldots, \lambda_{\xi_{\ell,P}}(t))\}_{\ell=1}^L, (\lambda_{u_1}(t), \ldots, \lambda_{u_P}(t))\right).$$

formally satisfy the backward dynamics

$$
\begin{cases}
-\dot{\lambda}(t) = \left(\partial_{X(t)}\mathcal{F}(X(t), \Upsilon)\right)^\top \lambda(t), \quad t \in (0, T) \setminus \mathbb{T} \\
\lambda(T) = \partial_{X(T)}\left(\dfrac{1}{N}\displaystyle\sum_{i=1}^N \Psi_i\right) \\
\text{Adjoint jump condition}
\end{cases}
\tag{5.5}
$$

where for all $i \in [\![N]\!]$ we have denoted

$$\Psi_i = \text{loss}\left(\mathcal{R}(\boldsymbol{u}(T, \boldsymbol{x}_i), y_i\right).$$

The ''adjoint jump condition'' in (5.5) captures how the discontinuous reset of the forward dynamics at spike times affects the backward propagation of sensitivities. Intuitively, whenever the membrane potential is reset in the forward system (5.4), the adjoint variables must also be updated to account for the sudden change in the state trajectory. This ensures that the contribution of each spike event to the overall objective functional is correctly transmitted backward in time. In other words, the jump condition enforces consistency between the discontinuous forward evolution and the backward sensitivity propagation, guaranteeing that the computed gradient faithfully reflects the effect of each reset event on the final loss. However, explicitly handling these jumps makes the adjoint equations cumbersome and the resulting gradients difficult to use in practice. This motivates the introduction of a mollification strategy to smooth out the discontinuities and eliminate the need for explicit jump conditions.

21

Concretely, let $\zeta > 0$ and consider the smooth reset mollifier

$$H_\zeta : \mathbb{R} \to (0,1), \quad H_\zeta(s) := \frac{1}{2}\left(1 + \tanh\left(\frac{\zeta s}{2}\right)\right) \tag{5.6}$$

and the (component-wise) discharge map

$$D_\zeta(s;\theta) := \left(1 - H_\zeta(s-\theta)\right)s. \tag{5.7}$$

Notice that, as $\alpha \to +\infty$, we have the point-wise convergence

$$H_\zeta(s) \to H(s) = \begin{cases} 1 & \text{if } s > 0 \\ \frac{1}{2} & \text{if } s = 0 \\ 0 & \text{if } s < 0 \end{cases} \quad \text{and} \quad D_\zeta(s;\theta) \to D(s;\theta) = \begin{cases} s & \text{if } s > \theta \\ \frac{\theta}{2} & \text{if } s = \theta \\ 0 & \text{if } s < \theta \end{cases}$$

Then, for the input neuron v with threshold θ_v and for each hidden neuron $\xi_{\ell,p}$ with threshold $\theta_{\ell,p}$ and current $J_\ell(t)$ as in (2.1), consider the mollified dynamics

$$\begin{cases} \dot{v}(t) = \frac{1}{\tau_v}\left(-v(t) + \langle \boldsymbol{a}, \boldsymbol{x}_i\rangle\right), & t \in (0,T) \setminus \mathbb{T}_0 \\ v(0) = 0, \\ v(t^{*+}) = D_\zeta\left(v(t^{*-}); \theta_v\right), & \text{for all } t^* \in \mathbb{T}_0 \end{cases} \tag{5.8}$$

and

$$\begin{cases} \dot{\xi}_{\ell,p}(t) = \frac{1}{\tau_{\ell,p}}\left(-\xi_{\ell,p}(t) + \omega_{\ell,p}J_\ell(t)\right), & t \in (0,T) \setminus \mathcal{T}_{\ell,p} \\ \xi_{\ell,p}(0) = 0, \\ \xi_{\ell,p}(t^{*+}) = D_\zeta\left(\xi_{\ell,p}(t^{*-}); \theta_{\ell,p}\right), & \text{for all } t^* \in \mathcal{T}_{\ell,p} \end{cases} \tag{5.9}$$

With these mollifications (5.8) and (5.9), the flow induced by $\mathcal{F}$ is differentiable in X almost everywhere and admits well-defined Jacobians across events. Consequently, the stacked adjoint is given in the classical sense by

$$\begin{cases} -\dot{\lambda}(t) = \left(\partial_{X(t)}\mathcal{F}(X(t), \Upsilon)\right)^\top \lambda(t), & t \in (0,T) \\ \lambda(T) = \partial_{X(T)}\left(\frac{1}{N}\sum_{i=1}^{N}\Psi_i\right) \end{cases}$$

and all gradients with respect to Υ can be computed directly by automatic differentiation through the (discretized) forward solver. In the limit $\zeta \to +\infty$, the mollified dynamics recover the original hybrid model (see Section 5.1.1) while retaining numerically stable gradients for training.

Finally, let us stress that there is no need of introducing a mollified version of the output neurons' dynamics, since in (Out) we have introduced no reset mechanism and the solution u_p is smooth over the whole time interval $(0,T)$.

5.1.1. *Convergence of the mollified dynamics.* For completeness, we analyze here the convergence of the mollified dynamics (5.8) and (5.9) to their hybrid counterparts (In) and (Hidd).

Proposition 5.1. *Let $T > 0$ and fix $\boldsymbol{a} \in \mathbb{R}^d$, $\tau_v > 0$, and $\theta_v > 0$. Let v denote the solution of* (In) *with spike times set $\mathbb{T}_0 = \{t_{0,k}\}_{k=1}^{K}$ characterized in Proposition 3.1. We assume all spikes are transversal threshold crossings, i.e. $\dot{v}(t_{0,k}) \neq 0$ for all $k \in [\![K]\!]$. Let $\{H_\zeta\}_{\zeta>0}$ be a family of smooth reset mollifiers $H_\zeta : \mathbb{R} \to (0,1)$ defined as in* (5.6). *For each $\zeta > 0$ define the discharge map $D_\zeta(z;\theta)$ as in* (5.7) *and let v_ζ be the solution to the reset–mollified input dynamics* (5.8). *Then*

$$\lim_{\zeta \to +\infty} \sup_{t \in (0,T)} |v_\zeta(t) - v(t)| = 0,$$

that is, $v_\zeta \to v$ uniformly on $(0,T)$.

Proof. First of all, let us notice that, if $\langle \boldsymbol{a}, \boldsymbol{x}_i \rangle \le \theta_v$, then we have from Proposition 3.1 that $\mathbb{T}_0 = \varnothing$ and, clearly, $v \equiv v_\zeta$ for all $t \in (0, T)$.

Assume now that $\langle \boldsymbol{a}, \boldsymbol{x}_i \rangle > \theta_v$. We shall proceed through two steps.

Step 1: agreement of flows between spikes. Let $\mathbb{T}_0 = \{t_{0,k}\}_{k=1}^K \subset (0, T)$ as in Proposition 3.1. On each subinterval $[t_{0,k}, t_{0,k+1})$, both v and v_ζ solve the same linear ODE with (possibly) different initial data at $t_{0,k}^+$. Hence, for $t \in [t_{0,k}, t_{0,k+1})$, we have

$$v_\zeta(t) - v(t) = \big(v_\zeta(t_{0,k}^+) - v(t_{0,k}^+)\big) e^{-\frac{t - t_{0,k}}{\tau_v}}. \tag{5.10}$$

Now, at each spike time $t_{0,k}$, we have

$$v(t_{0,k}^+) = 0 \quad \text{and} \quad v_\zeta(t_{0,k}^+) = \big(1 - H_\zeta(v_\zeta(t_{0,k}^-) - \theta_v)\big) v_\zeta(t_{0,k}^-).$$

Moreover, we also know that $v_\zeta(t_{0,k}^-) \to v(t_{0,k}^-) = \theta_v$ and, because the threshold crossing in the hybrid system is transversal $(\dot{v}(t_{0,k}) \ne 0)$, for sufficiently large ζ, the mollified trajectory approaches the threshold from the super-threshold side: $v_\zeta(t_{0,k}) - \theta_v > 0$. Consequently,

$$H_\zeta\big(v_\zeta(t_{0,k}^+) - \theta_v\big) \to 1 \quad \text{as } \zeta \to +\infty$$

and

$$\lim_{\zeta \to +\infty} v_\zeta(t_{0,k}^+) = \lim_{\zeta \to +\infty} \big(1 - H_\zeta(\cdot)\big) v_\zeta(t_{0,k}^-) = 0 = v(t_{0,k}^+).$$

Therefore,

$$e_{\zeta,k} := |v_\zeta(t_{0,k}^+) - v(t_{0,k}^+)| \to 0 \text{ as } \zeta \to +\infty \text{ for every } k \in [\![K]\!]. \tag{5.11}$$

Combining (5.10) and (5.11) yields, for all $t \in [t_{0,k}, t_{0,k+1})$,

$$\sup_{t \in [t_{0,k}, t_{0,k+1})} |v_\zeta(t) - v(t)| \le e_{\zeta,k} \, e^{-\frac{t - t_{0,k}}{\tau_v}} \le e_{\zeta,k}.$$

Taking the supremum over k finally gives

$$\sup_{t \in (0,T) \setminus \mathbb{T}_0} |v_\zeta(t) - v(t)| \le \max_{k \in [\![K]\!]} e_{\zeta,k} \to 0 \quad \text{as } \zeta \to +\infty.$$

Thus $v_\zeta \to v$ uniformly. $\qquad\square$

Having established the uniform convergence for the input neuron's mollified dynamics (5.8), we now pass to the hidden layers. The mechanism is entirely parallel: between spikes the trajectories follow the same linear RC flow (now driven by the presynaptic current J_ℓ), and the only approximation occurs at resets through the same discharge map D_ζ. Consequently, the time-change argument and the control of post-reset mismatches carry over verbatim, neuronwise and layerwise. For completeness and notational continuity, we state next the hidden-layer analogue; its proof is identical in structure to the input-layer case and is therefore omitted.

Proposition 5.2. *Let $T > 0$ and for given $\ell \in [\![L]\!]$ and $p \in [\![P]\!]$ fix $\omega_{\ell,p} \in \mathbb{R}$, $\tau_{\ell,p} > 0$ and $\theta_{\ell,p} > 0$. Let $\xi_{\ell,p}$ denote the solution of (Hidd) with spike times set $\mathcal{T}_{\ell,p}$. We assume all spikes are transversal threshold crossings, i.e. $\dot{\xi}_{\ell,p}(t^*) \ne 0$ for all $t^* \in \mathcal{T}_{\ell,p}$. Let $\{H_\zeta\}_{\zeta>0}$ be a family of smooth reset mollifiers $H_\zeta : \mathbb{R} \to (0, 1)$ defined as in (5.6). For each $\zeta > 0$ define the discharge map $D_\zeta(z; \theta)$ as in (5.7) and let $\xi_{\zeta,\ell,p}$ be the solution to the reset–mollified hidden dynamics (5.9). Then*

$$\lim_{\zeta \to +\infty} \sup_{t \in (0,T)} |\xi_{\zeta,\ell,p}(t) - \xi_{\ell,p}(t)| = 0,$$

that is, $\xi_{\zeta,\ell,p} \to \xi_{\ell,p}$ uniformly on $(0, T)$.

The uniform convergence of trajectories established in Propositions 5.1 and 5.2 guarantees that, away from spike events, the mollified solutions shadow the hybrid ones arbitrarily closely as $\zeta \to +\infty$. However, for spiking systems the most relevant qualitative feature is not the continuous evolution between resets but the precise timing of threshold crossings. To pass from convergence of trajectories to convergence of spike trains, one must ensure that the times at which the mollified trajectories hit threshold converge to the spike times of the hybrid system. This is not automatic, since a uniform approximation of the state does not by itself prevent small displacements in the moments of threshold crossing. The key property that makes this possible is the transversality of crossings in the hybrid system: at each spike time, the potential crosses the threshold with nonzero slope.

We emphasize that assuming transversality of threshold crossings is both natural and non-restrictive. In fact, transversality simply requires that the membrane potential crosses the threshold with nonzero velocity, which corresponds precisely to the operational definition of a spike in spiking neuron models. Non-transversal events, where the trajectory grazes the threshold tangentially, arise only under finely tuned parameter configurations and thus form a measure-zero set. Excluding them does not limit the generality of the analysis and ensures that spike times are well-defined and stable under perturbations. Under this assumption, standard perturbation arguments (implicit function theorem) ensure that the mollified trajectories admit corresponding threshold crossings whose timings depend continuously on ζ. This observation motivates the following proposition.

Proposition 5.3. *Under the assumptions of Proposition 5.1 (resp., 5.2), let $\{t_k\}$ denote the spike times of the hybrid system (In) (resp., (Hidd)). Then, for each t_k^* there exists a spike time $t_{\zeta,k}$ of the ζ-mollified system such that*

$$t_{\zeta,k} \to t_k \quad \text{as } \zeta \to +\infty.$$

Moreover, the entire spike train of the mollified system converges to that of the hybrid system.

Proof. For brevity, we treat only the input neuron; the argument for hidden neurons is identical and left to the reader.

Fix a spike time t_k of the hybrid system (In), i.e. $v(t_k^-) = \theta_v$ and $v(t_k^+) = 0$. By assumption, this threshold crossing is transversal, meaning $\dot{v}(t_k^-) \neq 0$.

From Proposition 5.1, we know that $v_\zeta \to v$ uniformly on $(0, T)$. In particular, in a neighborhood $[t_k - \delta, t_k + \delta]$, we have

$$\sup_{t \in [t_k - \delta, t_k + \delta]} |v_\zeta(t) - v(t)| \to 0 \quad \text{as } \zeta \to +\infty.$$

Thus, for large ζ, the trajectory v_ζ remains uniformly close to v in this neighborhood.

Now, because $\dot{v}(t_k^-) \neq 0$, there exists $\delta > 0$ such that $v(t) < \theta_v$ for $t < t_k - \delta$ and $v(t) > \theta_v$ for $t \in (t_k - \delta, t_k)$ (or the opposite inequality, depending on the sign of the slope). Uniform convergence then implies that for large enough ζ, v_ζ must also cross the threshold θ_v somewhere in $(t_k - \delta, t_k + \delta)$. Define this crossing as $t_{\zeta,k}$ and consider the smooth function

$$h_\zeta(t) := v_\zeta(t) - \theta_v.$$

We have $h_\zeta(t_{\zeta,k}) = 0$. Moreover, by construction, $h_\zeta \to h(t) = v(t) - \theta_v$ uniformly near t_k and $\dot{h}(t_k^-) = \dot{v}(t_k^-) \neq 0$. For large ζ, $\dot{h}_\zeta$ remains close to $\dot{h}$, so it does not vanish in a small neighborhood of t_k. Hence the implicit function theorem guarantees that the zero of h_ζ in that neighborhood is unique and depends continuously on ζ.

Because $t_{\zeta,k} \in (t_k - \delta, t_k + \delta)$ by construction, this continuity of the zero w.r.t. uniform perturbations then implies that

$$t_{\zeta,k} \to t_k \quad \text{as } \zeta \to +\infty.$$

24

Finally, since the hybrid system has finitely many isolated spikes, the argument can be repeated for each t_k. Thus the entire spike train of the mollified system converges to that of the hybrid system, with each spike time converging individually. □

6. Simulation experiments

To complement the theoretical contributions developed in the preceding sections, we present here a set of simulation experiments designed to illustrate the performance of the proposed spiking neural network (SNN) model on a benchmark classification problem. Although intentionally simple, these experiments serve to validate our framework and to highlight the expressive and computational properties of the architecture in practice.

All experiments presented in this section, together with the corresponding code, are available on GitHub [3].

6.1. **Dataset and architecture description.** We employ the `make_moons` dataset from the `scikit-learn` library, a widely used synthetic dataset for binary classification. The data consist of two interleaving half-circles embedded in the plane, which form a nonlinear decision boundary that cannot be captured by linear classifiers. Points are generated by sampling from these two half-moon manifolds and assigning binary labels according to membership.

The dataset can be enriched with Gaussian noise, thereby introducing variability and overlap between the classes. In this spirit, we considered two variants (see Figure 6.1): one with 0% noise, where the classes are perfectly separable, and one with 20% noise, where a significant fraction of the samples are perturbed, yielding overlapping distributions. The first variant represents a clean classification setting, while the second one provides a more realistic and challenging scenario that allows us to assess robustness to noise. Moreover, in both scenarios, we have split the dataset among 70% training data, 5% validation data, and 25% testa data.
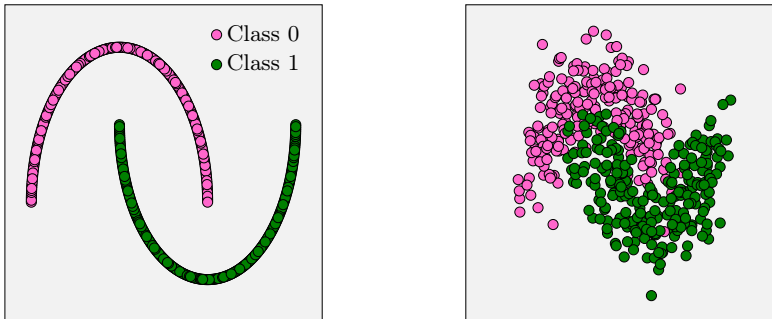

FIGURE 6.1. Training dataset with 0% noise (left) and 20% noise (right).

Although the `make_moons` dataset is simple compared to real-world data, it is well suited to our purposes. The main goal of this paper is not to propose large-scale benchmarks, but rather to investigate the mathematical underpinnings of SNN architectures, including their universal approximation properties and spike-trend dynamics. For this reason, a simple yet nonlinear dataset suffices to demonstrate the validity and potential of our framework.

As for the model's architecture, we employ a single hidden layer SNN ($L = 1$) with $P = 8$ neurons. Taking into account the input dimension $d = 2$, this amounts at a total number of only 19 trainable parameters. The architecture follows the dynamical modeling described in Sections 2 and 5 of this paper, combining (possibly mollified) LIF dynamics in the input and hidden layers with a static readout for classification. The choice of a relatively small network is deliberate: given the limited complexity of the `make_moons` dataset, adding depth or width would not necessarily improve

performance. On the contrary, using a compact architecture demonstrates that our model achieves competitive results with minimal resources, highlighting the efficiency of the proposed design.

Finally, we have considered the following structural hyper-parameters for the LIF dynamics in our SNN architecture:

Time horizon: $T = 60s$
Input layer: $\tau_v = 8$ and $\theta_v = 0.8$
Hidden layers: $\tau_{\ell,p} = 6$ and $\theta_{\ell,p} = 0.25$ for all $\ell \in [\![L]\!]$ and $p \in [\![P]\!]$
Output layer: $\tau_u = 10$ and $\theta_u = 0.3$
Gaussian mollification: $\mu = 0.2$

As for the mollification hyperparameter ζ, its value is not fixed but follows a smooth geometric schedule across training. Specifically, ζ is initialized at a low value $\zeta_0 = 3$ and gradually increased to $\zeta_1 = 10$ over the course of training epochs, according to the update rule

$$\zeta(e) = \zeta_0 \left(\frac{\zeta_1}{\zeta_0}\right)^{\frac{e}{E-1}},$$

where e is the epoch index and E the total number of epochs. This choice yields a progressive sharpening of the surrogate gradient while maintaining stability in the early stages.

6.2. **Training and evaluation metrics.** Training is performed via surrogate-gradient optimization, following the regularized empirical risk minimization framework introduced in Section 5. Since we are considering a binary classification problem, we have chosen the binary cross entropy as training loss function. To evaluate performance, we consider four standard classification metrics:

Accuracy, measuring the overall proportion of correct predictions;
Precision, assessing the reliability of positive predictions;
Recall, measuring the ability to correctly identify all positive samples;
F1-score, the harmonic mean of precision and recall, providing a single measure that balances the trade-off between false positives and false negatives.

Simulations have been performed in `Python` on a Acer TravelMate P414-53 laptop with OS Ubuntu 24.04.3 LTS, 13th Gen Intel® Core™ i5-1335U × 12 processor and 32 GiB of RAM memory.

We report results on training, validation, and test splits, in order to assess generalization. On the dataset with 0% noise, the model achieves accuracy, precision, and F1-score in the range 88%–92%, consistently across training, validation, and test sets. The recall metric is slightly lower, around 84%, reflecting a mild tendency of the classifier to miss some positive samples (see Figure 6.2 and Table 6.1). Overall, these results confirm that the proposed SNN successfully captures the nonlinear decision boundary of the dataset while maintaining generalization.
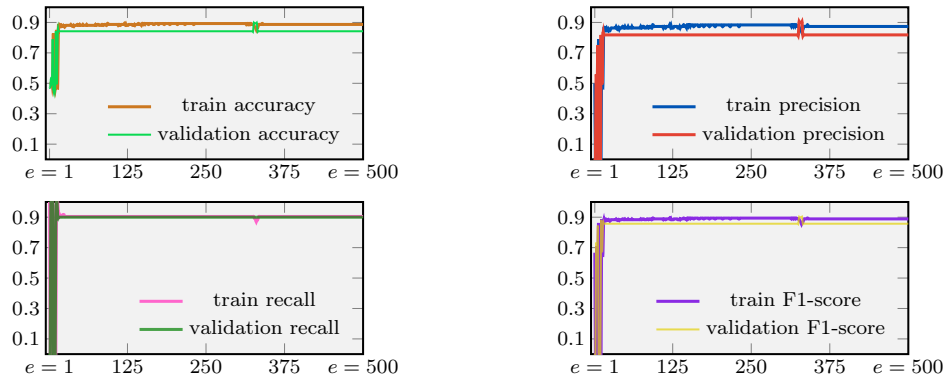

FIGURE 6.2. Training and validation metrics on the dataset with 0% noise.

26

	Accuracy	Precision	Recall	F1-score
0% noise	0.8889	0.9286	0.8387	0.8814
20% noise	0.8413	0.8889	0.7742	0.8276

TABLE 6.1. Test metrics obtained by our SNN on the `make_moons` dataset with 0% and 20% noise.

On the dataset with 20% noise, performance understandably decreases, with metrics lying in the range 75%–88% (see Figure 6.3 and Table 6.1). Among them, recall is again the lowest, reflecting the greater difficulty of recovering true positives in the presence of noisy, overlapping samples. Nonetheless, precision and accuracy remain close to 90%, indicating that the model preserves a strong discriminative ability even under perturbations.
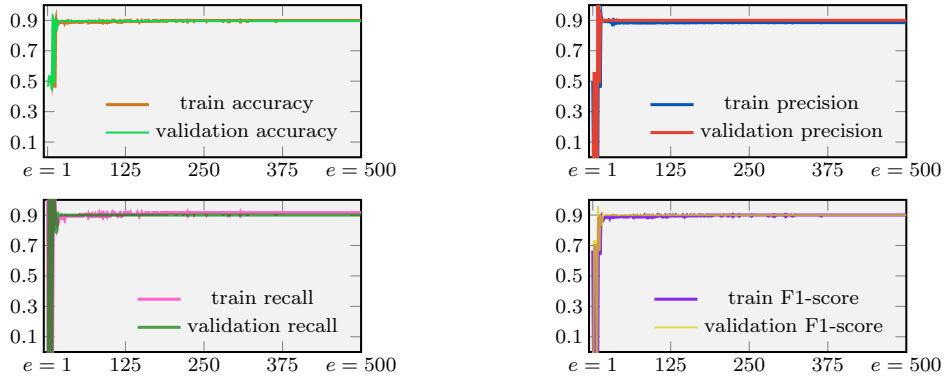

FIGURE 6.3. Training and validation metrics on the dataset with 20% noise.

We shall also stress that, in both experimental scenarios we have considered, training and validation metrics are systematically higher than those obtained on the test set (around 90% versus slightly lower values on test). This behavior is natural: since the test set contains previously unseen samples, including challenging points near the nonlinear boundary or affected by noise, slightly reduced performance is expected. Importantly, the gap remains modest, showing that the model does not suffer from significant overfitting.

These results support several key conclusions. First, they confirm that even a small SNN – comprising only a single hidden layer with 8 neurons – can efficiently solve nonlinear classification tasks, in line with the universal approximation principles established in Section 4. Second, the experiments show that the model remains robust to moderate levels of noise, maintaining good performance across metrics. Although recall drops both in the clean and noisy regimes, this effect is expected and does not compromise the overall effectiveness of the classifier. Indeed, the fact that accuracy and precision remain high underscores the stability and reliability of the spiking dynamics. Finally, the observation that training and validation metrics are consistently higher than test ones is not a drawback but rather an indication of good generalization: the model maintains strong performance even on unseen and more difficult data.

Overall, these experiments, while intentionally simple, validate the theoretical analysis developed throughout the paper. They demonstrate that the proposed SNN architecture is not only mathematically sound but also effective in practice, even under constrained architectures and noisy conditions.

7. Conclusions and open problems

In this work we have developed a rigorous theoretical framework for Spiking Neural Networks (SNNs), focusing on two central aspects: their expressive power and the dynamical constraints that govern spike transmission. By proving a constructive universal approximation theorem for SNNs built from Leaky Integrate-and-Fire (LIF) neurons with threshold-reset dynamics, we established that narrow spiking architectures can approximate any continuous function on compact domains to arbitrary accuracy. Complementing this, our analysis of spike propagation across layers revealed a general non-increasing trend in spike counts, with precise criteria for the exceptional regimes – such as overlapping presynaptic spikes and high gain-to-threshold ratios – where spike counts may increase. Taken together, these results offer a principled mathematical foundation for understanding both the power and the limitations of SNNs, providing guarantees that support their role in classification, signal processing, and beyond.

At the same time, our findings open several new directions for further research. We conclude by highlighting a number of open problems which we believe are particularly promising:

1. **Depth–width tradeoffs in SNNs.** While our universality result is constructive, it does not quantify the minimal number of neurons per layer or the necessary depth for approximation within a given error tolerance. Establishing precise depth–width tradeoffs, analogous to those known for classical ANNs and ResNets [1], would provide sharper guidelines for designing efficient spiking architectures.

2. **Quantitative approximation rates.** Our universal approximation Theorem 4.1 guarantees density in the space of continuous functions but does not provide explicit approximation rates in terms of the number of neurons, spikes, or layers. Deriving such rates would enable comparisons between SNNs and traditional neural networks and clarify whether temporal coding confers efficiency advantages.

3. **Training under hybrid dynamics.** The introduction of mollifiers in the LIF dynamics ensures well-posedness and differentiability, but the resulting surrogate models are only approximations of true spiking dynamics. A central challenge is to design principled training algorithms that directly exploit the hybrid nature of spike-reset systems while still yielding tractable gradients, perhaps by combining tools from optimal control and non-smooth analysis.

4. **Generalization and capacity bounds.** Beyond approximation, little is known about the generalization properties of SNNs. Establishing statistical learning bounds – such as Rademacher complexity [2] or VC-dimension [31] estimates tailored to spike-based models – remains an open question with immediate implications for their reliability in practical tasks.

5. **Dynamics beyond LIF models.** Our analysis focused on LIF neurons, but biologically realistic neurons exhibit richer dynamics (e.g., adaptive thresholds, conductance-based models, or stochastic firing). Extending universal approximation results and dynamical analyses to these broader families would deepen the connection between theoretical SNNs and neuroscience.

In summary, this work provides a foundational step toward a mathematically rigorous theory of SNNs. Addressing the open problems listed above will be crucial to bridge the gap between their theoretical expressiveness and practical performance, ultimately advancing both neuromorphic computation and our understanding of biological neural systems.

References

[1] A. Alvarez-Lopez, A. H. Slimane, and E. Zuazua. Interplay between depth and width for interpolation in neural ODEs. *Neural Netw.*, 180:106640, 2024.

[2] P. L. Bartlett and S. Mendelson. Rademacher and Gaussian complexities: risk bounds and structural results. *J. Mach. Learn. Res.*, 3(Nov):463--482, 2002.

[3] U. Biccari. https://github.com/umbertoBiccari/Spiking-Neural-Networks, 2025.

[4] M. Bouvier, A. Valentian, T. Mesquida, F. Rummens, M. Reyboz, E. Vianello, and E. Beigne. Spiking neural networks hardware implementations and challenges: A survey. *ACM J. Emerg. Technol. Comput. Syst.*, 15(2):1--35, 2019.

[5] G. Cybenko. Approximation by superpositions of a sigmoidal function. *Math. Control Signals Syst.*, 2(4):303--314, 1989.

[6] P. Dayan and L. F. Abbott. *Theoretical neuroscience: computational and mathematical modeling of neural systems*. MIT press, 2005.

[7] R. A. DeVore and G. G. Lorentz. *Constructive approximation*, volume 303. Springer Science & Business Media, 1993.

[8] J. K. Eshraghian, M. Ward, E. O. Neftci, X. Wang, G. Lenz, G. Dwivedi, M. Bennamoun, D. S. Jeong, and W. D. Lu. Training Spiking Neural Networks using lessons from deep learning. *IEEE Proc.*, 111(9):1016--1054, 2023.

[9] K.-i. Funahashi and Y. Nakamura. Approximation of dynamical systems by continuous time recurrent neural networks. *Neural Netw.*, 6(6):801--806, 1993.

[10] W. Gerstner and W. M. Kistler. *Spiking neuron models: Single neurons, populations, plasticity*. Cambridge university press, 2002.

[11] W. Gerstner, W. M. Kistler, R. Naud, and L. Paninski. *Neuronal dynamics: From single neurons to networks and models of cognition*. Cambridge University Press, 2014.

[12] A. Haar. Zur theorie der orthogonalen funktionensysteme. *Math. Ann.*, 71(1):38--53, 1911.

[13] A. L. Hodgkin and A. F. Huxley. A quantitative description of membrane current and its application to conduction and excitation in nerve. *J. Physiol.*, 117(4):500, 1952.

[14] C. Koch. *Biophysics of computation: information processing in single neurons*. Oxford university press, 2004.

[15] V. Lakshmikantham, D. D. Bainov, and P. Simeonov. *Theory of impulsive differential equations*, volume 6. World scientific, 1989.

[16] L. Lapicque. Recherches quantitatives sur l'excitation electrique des nerfs. *J. Physiol. and Pathology*, 9:620--635, 1907.

[17] S. B. Laughlin. Energy as a constraint on the coding and processing of sensory information. *Curr. Opin. Neurobiol.*, 11(4):475--480, 2001.

[18] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *nature*, 521(7553):436--444, 2015.

[19] M. Leshno, V. Y. Lin, A. Pinkus, and S. Schocken. Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. *Neural Netw.*, 6(6):861--867, 1993.

[20] R. Llinás. Thalamocortical assemblies: How ion channels, single neurons and large-scale networks organize sleep oscillations. *Thalamus Relat. Syst.*, 2(1):87--88, 2002.

[21] W. Maass. Lower bounds for the computational power of networks of spiking neurons. *Neural Comput.*, 8(1):1--40, 1996.

[22] W. Maass. Networks of spiking neurons: the third generation of neural network models. *Neural Netw.*, 10(9):1659--1671, 1997.

[23] Y. Meyer. *Wavelets and operators*. Number 37. Cambridge university press, 1992.

[24] J. D. Nunes, M. Carvalho, D. Carneiro, and J. S. Cardoso. Spiking neural networks: A survey. *IEEE access*, 10:60738--60764, 2022.

[25] N. Perez-Nieves and D. Goodman. Sparse spiking gradient descent. *Adv. Neural Inf. Process. Syst.*, 34:11795--11808, 2021.

[26] M. Pfeiffer and T. Pfeil. Deep learning with spiking neurons: Opportunities and challenges. *Front. Neurosci.*, 12:409662, 2018.

[27] L. Pontryagin. The mathematical theory of optimal processes. *Interscience*, 1962.

[28] E. D. Sontag. *Mathematical control theory: deterministic finite dimensional systems*, volume 6. Springer Science & Business Media, 2013.

[29] A. Tavanaei, M. Ghodrati, S. R. Kheradpisheh, T. Masquelier, and A. Maida. Deep learning in spiking neural networks. *Neural Netw.*, 111:47--63, 2019.

[30] E. Trélat. *Contrôle optimal: théorie & applications*, volume 36. Vuibert Paris, 2005.

[31] V. N. Vapnik. An overview of statistical learning theory. *IEEE Trans. Neur. Netw.*, 10(5):988--999, 1999.

* Chair of Computational Mathematics, DeustoTech, University of Deusto, Avenida de las Universidades 24, 48007 Bilbao, Basque Country, Spain

Email address: umberto.biccari@deusto.es